Computational Modeling of Lexical Ambiguity

Linlin Li

Computer Science

Saarland University

thesis for obtaining the title of Doctor of Natural Sciences of
the Faculties of Natural Sciences and Technology of Saarland University

First Reviewer: Dr. Caroline Sporleder

Second Reviewer: Prof. Dr. Manfred Pinkal

Committee Chair: Prof. Dr. Gerhard Weikum

Committee Member: Dr. Ivan Titov

Dean of Faculty: Prof. Dr. Mark Groves

Day of the defense: October 10, 2012

Abstract

Lexical ambiguity is a frequent phenomenon that can occur not only for words but also on the phrase level. Natural language processing systems need to efficiently deal with these ambiguities in various tasks, however, we often encounter such system failures in real applications. This thesis studies several complex phenomena related to word/phrase ambiguity at the level of text and proposes computational models to tackle these phenomena.

Throughout the thesis, we address a number of lexical ambiguity phenomena varying across the sense granularity line. We start with the idiom detection task, in which candidate senses are constrained to'literal' and 'idiomatic'. Then, we move on to the more general case of detecting figurative expressions. In this task, target phrases are not lexicalized but rather bear nonliteral semantic meanings. Similar to the idiom task, this one has two candidate sense categories ('literal' and 'nonliteral'). Next, we consider a more complicated situation where words often have more than two candidate senses and the sense boundaries are fuzzier, namely word sense disambiguation (WSD). Finally, we discuss another lexical ambiguity problem in which the sense inventory is not explicitly specified, word sense induction (WSI).

Computationally, we propose novel models that outperform state-of-the-art systems. We start with a supervised model in which we study a number of semantic relatedness features combined with linguistically informed features such as local/-global context, part-of-speech tags, syntactic structure, named entities and sentence markers. While experimental results show that the supervised model can effectively detect idiomatic expressions, we further improve the work by proposing an unsupervised bootstrapping model which does not rely on human annotated data but performs at a comparative level to the supervised model. Moving on to accommodate other lexical ambiguity phenomena, we propose a Gaussian Mixture Model

that can be used not only for detecting idiomatic expressions but also for extracting unlexicalized figurative expressions from raw corpora automatically. Aiming at modeling multiple sense disambiguation tasks within a uniform framework, we propose a probabilistic model (topic model), which encodes human knowledge as sense priors via paraphrases of gold-standard sense inventories, to effectively perform on the idiom task as well as two WSD tasks. Dealing with WSI, we find state-of-the-art WSI research is hindered by the deficiencies of evaluation measures that are in favor of either very fine-grained or very coarse-grained cluster output. We argue that the information theoretic V-Measure is a promising approach to pursue in the future but should be based on more precise entropy estimators, supported by evidence from the entropy bias analysis, simulation experiments, and stochastic predictions. We evaluate all our proposed models against state-of-the-art systems on standard test data sets, and we show that our approaches advance the state-of-the-art.

Zusammenfassung

Lexikalische Mehrdeutigkeit ist ein häufiges Phänomen, das nicht nur auf Wort, sondern auch auf phrasaler Ebene auftreten kann. Systeme zur Verarbeitung natürlicher Sprache müssen diese Mehrdeutigkeiten in verschiedenen Aufgaben effizient bewältigen, doch in realen Anwendungen erweisen sich solche Systeme oft als fehlerhaft. Ziel dieser Dissertation ist es verschiedene komplexe Phänomene lexikalischer und insbesondere phrasaler Mehrdeutigkeit zu erforschen und algorithmische Modelle zur Verarbeitung dieser Phänomene vorzuschlagen.

In dieser Dissertation beschäftigen wir uns durchgehend mit einer Reihe von Phänomenen lexikalischer Ambiguität, die in der Granularität der Sinnunterschiede variieren: Wir beginnen mit der Aufgabe Redewendungen zu erkennen, in der die möglichen Bedeutungen auf 'wörtlich' und 'idiomatisch' beschränkt sind; dann fahren wir mit einem allgemeineren Fall fort in dem die Zielphrasen keine feststehenden Redewendungen sind, aber im Kontext eine übertragene Bedeutung haben. Wir definieren hier die Aufgabe bildhafte Ausdrücke zu erkennen als Disambiguierungs-Problem in der es, ähnlich wie in der Redewendungs-Aufgabe, zwei mögliche Bedeutungskategorien gibt ('wörtlich' und 'nicht-wörtlich').

Als nächstes betrachten wir eine kompliziertere Situation, in der Wörter oft mehr als zwei mögliche Bedeutungen haben und die Grenzen zwischen diesen Sinnen unschärfer sind, nämlich Wort-Bedeutungs-Unterscheidung (*Word Sense Disambiguation*, WSD); Schließlich diskutieren wir ein weiteres Problem lexikalischer Mehrdeutigkeit, in dem das Bedeutungsinventar nicht bereits ausdrücklich gegeben ist, d.h. Wort-Bedeutungs-Induktion (*Word Sense Induction*, WSI).

Auf algorithmischer Seite schlagen wir Modelle vor, die Systeme auf dem aktuellen Stand der Technik übertreffen. Wir beginnen mit einem überwachten Modell, in dem wir eine Reihe von Merkmalen basierend auf semantischer Ähnlichkeit mit linguistisch fundierten Merkmalen wie lokalem/globalem Kontext, Wortarten,

syntaktischer Struktur, Eigennamen und Satzzeichen kombinieren. Ausgehend von experimentellen Ergebnissen die zeigen, dass das überwachte Modell effektiv idiomatische Ausdrücke erkennen kann, verbessern wir unsere Arbeit indem wir ein unüberwachtes Bootstrapping-Modell präsentieren, das nicht auf manuell annotierte Daten angewiesen ist aber ähnlich gut funktioniert wie das überwachte Modell. Um weitere Phänomene lexikalischer Mehrdeutigkeit zu behandeln, schlagen wir des weiteren ein Gauss'sches Mischmodell vor, das nicht nur zur Erkennung von Redewendungen verwendet werden kann, sondern auch dazu effektiv und automatisch neue produktive bildhafte Ausdrücke aus unverarbeiteten Corpora zu extrahieren. Mit dem Ziel mehrere Aufgaben zur Disambiguierung innerhalb eines einheitlichen Systems zu modellieren, schlagen wir ein statistisches Modell (Topic-Modell) vor, um sowohl die Aufgabestellung der Redewendungs-Erkennung als auch die WSD-Probleme effektiv zu bearbeiten. Die A-priori-Wahrscheinlichkeiten dieses Modells kodieren menschliches Wissen, wozu es Gold-Standard-Bedeutungslexika benutzt. Bezüglich WSI stellen wir fest, dass der Stand der WSI-Forschung durch inadequate Evaluationsmaße behindert wird, die entweder sehr feinkörnige oder sehr grobkörnige Cluster-Ergebnisse bevorzugen. Wir behaupten, dass das Informationstheoretische 'V-Measure' ein vielversprechender Ansatz ist, der zukünftig verfolgt werden könnte, der jedoch mit präzieseren Entropie-Schätzern, unterstützt von Belegen aus der Entropie-Trend-Analyse, Simulationxexperimenten und stochastische Vorhersagen, aufbauen sollte.

Wir evaluieren alle unsere vorgeschlagenen Modelle auf standardisierten Testdaten und vergleichen sie mit anderen Systemen auf dem aktuellen Forschungsstand, und wir zeigen dass unsere Ansätze den aktuellen Forschungsstand voranbringen.

Acknowledgements

This thesis would not have been possible without the help from various people and organizations. First, I would like to thank my advisors. My first advisor, Caroline Sporleder, encouraged me to start this PhD program and has given me so much help and support throughout my research work. I'd like to give my heartfelt gratitude to her first. My other advisor, Manfred Pinkal, who has set a very friendly academic environment where I am lucky to have met and worked with great researchers, has always been helpful at pointing out directions whenever I went to him for advice.

My second thanks go to DFG (Deutsche Forschungsgemeinschaft), who have founded the Cluster of Excellence on "Multimodal Computing and Interaction" (MMCI) where I have worked as a research fellow for three years in the group headed by Dr. Sporleder. The generosity of the funding agency has made it possible for me to purely concentrate my time on the work in this thesis.

A number of people are less directly involved in this thesis but very important nonetheless. I thank Ivan Titov for pointing me to the direction of entropy bias, simulations and stochastic predictions during the study of word sense induction (WSI). Suresh Manandhar, Yannis Korkontzelos, and Ioannis Klapaftis have untiringly answered many of my questions on the settings and evaluation of the SemEval 2010 WSI shared task. Benjamin Roth has shared the trained topic model on the Wikipedia dump, saving me a lot of effort; he also made very helpful comments on the initial period of the Gaussian Mixture Model chapter. I am grateful to Xaver Koch and Todd Shore for annotating the UdSic and UdSfec corpora. I am also grateful to Daniel Bauer and Michaela Regneri for assisting me in translating the abstract into German (Zusammenfassung). During my PhD, I have bothered my former long-time office mate, Alexis Palmer, many times for paper proofreadings and English-language-related questions. I have also expectly or unexpectly visited

Yi Zhang's office hour many times not only for research-related questions but also experiment-related engineering questions.

I would like to thank all the people who have helped me to proofread the thesis and given constructive comments. They are Grzegorz Chrupala, Tomasz Jurkiewicz, Mateusz Malinowski, Alexis Palmer, Benjamin Roth, Ben Swanson and Yi Zhang.

I thank Tianfang Yao, who definitely deserves no less gratitude than anybody else on this list, for introducing me to the field of natural language processing, sending me to study in Germany, where I have gained valuable experiences not only professionally but also culturally.

Last but not least, I would like to thank my parents, 李学家 and 林志萍, whose positive attitude has set me an example since a very early stage of my life and shaped my basic attitude towards various situations, including handling the difficult parts of this thesis.

Contents

List of Figures

List of Tables

1

Introduction

We start this thesis by discussing an interesting lexical phenomenon that we noticed by comparing different cultural environments. Given a picture of a *full moon*, what comes to ones mind? In folklore and tradition, *full moon* is often associated with temporary insomnia, epilepsy, and various magical phenomena such as lycanthropy (Kelly et al., 1986). In contrast, the Chinese phrase 满月 *(full moon)* is often associated with family, love and share, which are frequently reflected in poems, e.g., "但愿人久长，千里共婵娟 *(wish us a long life so as to share the beauty of this graceful full-moon light, even thousands miles apart)*", "满月如璧 *(the jade like full moon)*", or cultural events, e.g., *the Moon Festival (August 15th of the Lunar Calendar), a traditional Chinese festival for family members getting together, sharing memory and thoughts, and eating Mooncake*. In fact, this association divergence is also captured by search engine results. When the English query keywords *full moon* are input to Google Image Search[1], pictures containing werewolf are among the top hits.[2], whereas images, illustrating the Chinese legend 嫦娥 *(the Princess of the Moon)* giving blessing to people on a full moon night, are among the typical associated pictures for the Chinese keywords. Actually, Chinese folk stories interpret the shadow of the mountains which are clearly visible on a full moon night as the figures of the Princess of the Moon and her pet 玉兔.

As it is suggested by this example, the complexity of wording meaning is often revealed in cross-linguistic analysis, where usually implicit assumptions and connotations are made explicit. Therefore, computational modeling of word meaning are faced with huge challenges. This thesis

[1] See `http://www.google.com/imghp`

[2] For copyright reasons, we cannot show these pictures here.

considers several complex phenomena related to word meaning at the level of text, and propose computational models to tackle these phenomena.

1.1 Motivation

In this section, we discuss practical factors that motivate our work. Lexical ambiguity[1] is a frequent phenomenon that can occur not only for words (Example 1.1[2]) but also on the phrase level (Example 1.2) in natural language. For instance, the word *plant* can mean "factory" (Example 1.1a), whereas it can also mean "a living organism" (Example 1.1b). While the English phrase *playing with fire* is often used idiomatically, which means *to take part in a dangerous or risky undertaking*[3] (Example 1.2a), it may also be used literally in some cases (Example 1.2b). Actually, in our study we find that 34 instances out of all 566 occurrences of *play with fire* in the English Gigaword[4] corpus are used literally (6%).

(1.1) (a) Germany's coalition government has announced a reversal of policy that will see all the country's nuclear power plants phased out by 2022.

(b) As of 2010, there are thought to be 300-315 thousand species of plants, of which the great majority, some 260-290 thousand, are seed plants.

(1.2) (a) Dissanayake said that Kumaratunga was "playing with fire" after she accused military's top brass of interfering in the peace process.

(b) Grilling outdoors is much more than just another dry-heat cooking method. It's the chance to play with fire, satisfying a primal urge to stir around in coals.

From the application side, Natural Language Processing (NLP) systems should effectively deal with lexical ambiguity in various tasks. However, we often encounter failures of such systems: Example 1.3 is an output by a machine translation system YAHOO! BABEL FISH[5], where the English idiomatic expression *spilled the beans*, which means "revealed the secret", is falsely literally translated into German.

[1] Throughout the thesis, we adopt a broad sense of lexical item, including both words and multi-word expressions (MWEs).

[2] Examples in this chapter are from real corpora.

[3] Defined by online dictionary `http://www.thefreedictionary.com/play+with+fire`

[4] See Section 2.1 for the description of the English Gigaword corpus

[5] See `http://babelfish.yahoo.com/`. The result was from the translation system in December 2008.

(1.3) (a) The government agent spilled the beans on the secret dossier.

(b) Der Regierungsbeauftragte verschüttete die Bohnen auf dem geheimen Dossier.

The frequent appearance of lexical ambiguity in natural language and the inefficiency of NLP systems dealing with such ambiguity call for advancement in the study of this topic. In this thesis, we aim to deal with the various lexical ambiguity phenomena from a computational modeling perspective and advance the state-of-the-art research.

1.2 Thesis Overview

We present several closely-related pieces of work in this thesis, varying along two lines: (i) The sense granularity line leads to research on different lexical ambiguity phenomena. As the chapters move forward, we aim at studying more complicated sense granularity problems. (ii) From a computational modeling point of view, we use more advanced models to improve the performance of state-of-the-art methods and reduce human annotation effort.

Sense Granularity Line We study four main types of tasks in this thesis: the distinction of 'literal'/'idiomatic' occurrences of potentially idiomatic expressions (Chapter 3 and 4), novel figurative expression detection (Chapter 5), Word Sense Disambiguation (WSD) (Chapter 6), and Word Sense Induction (WSI) (Chapter 7). We also combine the idiom task with the novel figurative expression task together and name the category as 'literal'/'idiomatic' MWE detection. The reason is that the two tasks are very similar in that both of them deal with whether a target expression is used literally or not, although the type of expressions that the two tasks study are different from each other.

In the 'literal' or 'nonliteral' MWE detection task, the problem is defined in a binary classification framework. In most of the cases, the literal and nonliteral readings are well separated (e.g., Figure 1.1[1]). As the semantics of the two readings are often clearly distinguishable from each other (e.g., *spill the beans* as "spill the beans onto the floor" v.s. *spill the beans* as "revealing the secret"), our computational models are able to achieve high performance on this type of tasks (see Chapter 3 and 4).

In contrast, more fine-grained sense categories are introduced in the WSD task. There are 10 different sense categories for the word *bank* as a noun in WordNet 2.1 (see Appendix A for a

[1]Figure 1.1, 1.2 and 1.3 are only for demonstration purpose. They do not reflect any true distribution of instances.

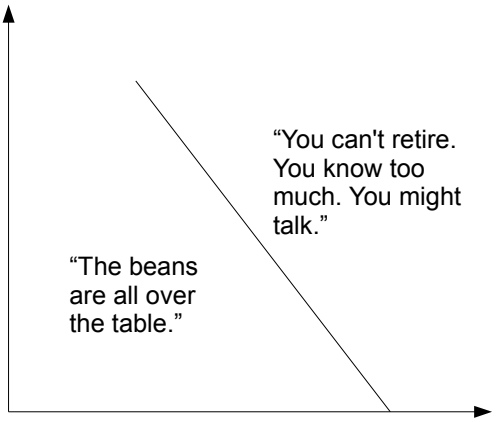


Figure 1.1: A demonstration example of a boundary of literal/nonliteral expressions. The 'literal' and 'nonliteral' readings of the phrase *spill the beans*.

complete set of sense category definitions for *bank* as defined in WordNet 2.1). If we project those categories in a coordinate plane, we are faced with much more complicated decision boundaries (e.g, Figure 1.2). The task of WSD poses a challenge due to the complexity of the sense inventories as well as the fact that some senses are difficult to tell apart in certain contexts (e.g., *A: "I need a loan." B: "go to the bank."*, *bank* as 'bank building' v.s. *bank* as 'financial institution').

WSI is even more challenging in that the sense inventories are not predefined. In WSI, instances are represented as clusters without the number of clusters being specified. For example, we draw decision boundaries of five categories in Figure 1.3 but this partition can be easily challenged by other types of decision boundaries if the system is required to output a cluster number other than five. In Chapter 7 we find that the study of WSI is negatively affected by the deficiencies of evaluation measures. Therefore, our study focuses on improving the current bottleneck (evaluation measures), and we believe that an advancement in WSI evaluation will eventually enhance the future development of this topic.

Statistical Modeling Line We investigate increasingly advanced models to reduce annotation effort (Chapter 4), increase the model performance (Chapter 5, 6), or boost the model performance with a minimum amount of extra knowledge (Chapter 6).

We develop a supervised model in Chapter 3, where we study different features and their combinations and evaluate the performance. Chapter 4 is built on top of the work of the previous

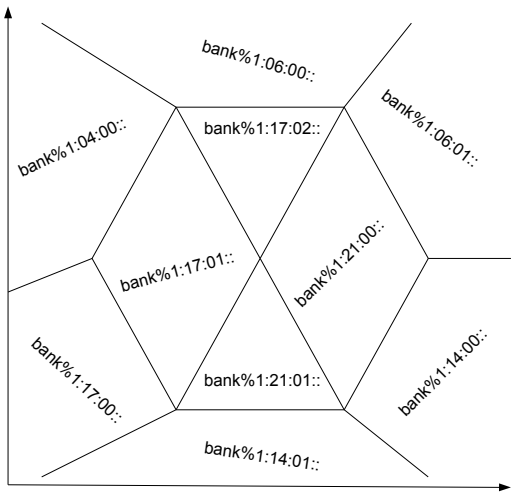


Figure 1.2: A demonstration example of decision boundaries of WSD. Different senses are represented by sense keys from WordNet 2.1, example word *bank*.

chapter, aiming to reduce the human annotation effort involved in labeling the gold standard instances. It proposes a bootstrapping model which utilizes two component classifiers from previous work. Although fully unsupervised, it maintains a performance comparable to the supervised approach. Chapter 6 introduces a probabilistic model (topic model) which encodes human knowledge as sense priors via paraphrase information. Experimental results show that while this additional information costs very little human work, it largely boosts the performance. Furthermore, the models developed in Chapter 6 can model multiple related tasks within one framework, which further reduces efforts involved.

The Gaussian Mixture Model (GMM) model proposed in Chapter 5 shares a number of semantic relatedness features with the models in Chapter 3 and Chapter 4. We demonstrate that similar features for related tasks may work for different statistical modeling approaches, thus effective selection of features is important. This GMM framework, which is partially built up on top of the supervised model features, can be effectively utilized to discover new nonliteral expressions.

We outline some problems of the V-Measure for WSI evaluation by carrying out entropy simulation experiments in Chapter 7, and propose alternative entropy estimators which can better serve the task. Furthermore, we adopt a novel approach, stochastic prediction, to accommodate weighted cluster output to the evaluation methodology.

The individual chapters are arranged as follows:

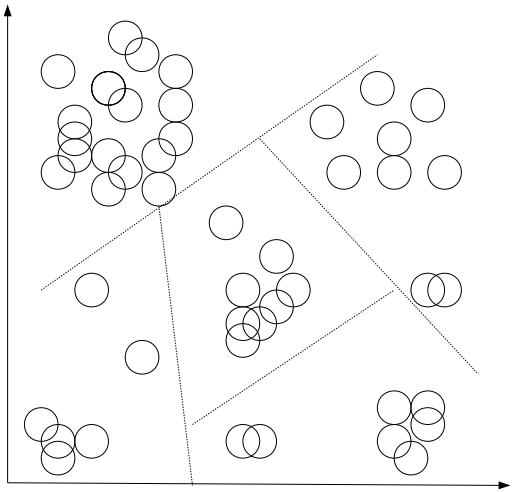


Figure 1.3: A demonstration example of decision boundaries of WSI. Different data examples are represented as circles, the number of senses is underspecified.

Chapter 3 proposes lexical cohesion based features, which is a continuation of our previous research (Li, 2008). The experiments show that lexical cohesion based features can be effectively utilized for idiom detection. Furthermore, these features can be complemented by bag-of-words features and more sophisticated linguistically informed features such as syntactic features, named entity features and sentence-marker features.

Chapter 4 aims to improve on the work of Chapter 3 by making the process fully unsupervised. We propose a bootstrapping framework based on two component classifiers from previous work, one unsupervised classifier (Li, 2008; Sporleder and Li, 2009) and one supervised classifier (Chapter 3). The bootstrapping framework is initialized by choosing the most confident examples from the unsupervised classifier and using them as the initial training set of the supervised classifier. Then the supervised classifier predicts label on the remaining examples, and the most confident examples are selected and added to the training set of the supervised classifier in the next round. The process iterates as the training set of the supervised classifier enlarges and the size of the unlabeled set shrinks, and it is terminated after a predefined number of iterations. We show that this strategy achieves very competitive results, even compared to the supervised strategy on the idiom detection task.

Chapter 5 While the previous two chapters focus on lexicalized idiomatic expression detection, Chapter 5 extends this topic further to unlexicalized figurative expressions (e.g., *spill the beans* v.s. *take the sock out of your mouth*). We propose a Gaussian Mixture Model (GMM)

for detecting novel figurative phrases in context. We evaluate our model on a small annotated dataset and show that this model outperforms a number of baseline systems. The parameters of the model can be estimated in an unsupervised way using the EM algorithm. Performance can be further improved by estimating the parameters from a small annotated data set.

Chapter 6 deals with not only two-sense ambiguity phenomena (the idiom detection task in Chapter 3 and 4 and the figurative expression detection task in Chapter 5), but also multiple-sense ambiguity phenomena (i.e., WSD). It models the two different types of tasks in a uniform framework, and presents a probabilistic model which chooses the best sense based on the conditional probability of sense paraphrases given a context. A topic model is introduced to decompose this conditional probability into two conditional probabilities with latent variables. We further propose three different instantiations of the model with different degrees of resource availability. The proposed models are tested on three different tasks: coarse-grained WSD, fine-grained WSD, and idiom detection. In all three cases, the models outperform state-of-the-art systems either quantitatively or statistically significantly.

Chapter 7 studies a sense ambiguity problem when the sense inventory is not predefined (WSI). We find that state-of-the-art WSI research is hindered by the deficiencies of evaluation, lacking a fair platform for comparison among different systems. We make two main findings: 1) The state-of-the-art supervised evaluation approach is strongly biased towards fine-grained sense category clusterings; 2) State-of-the-art unsupervised WSI evaluation approaches are in favor of either coarse-grained output (F-Measure) or fine-grained output (Entropy approach such as V-Measure). We discover that the entropy based evaluation approach uses a biased entropy estimator which leads to unreliable scoring, and propose alternative estimators to replace the state-of-the-art WSI evaluation entropy estimator. Furthermore, we also propose a solution to alleviate the entropy bias problem by constraining the number of clusters based on example size and number of gold-standard classes.

1.3 Main Contributions

In this thesis, we study various lexical ambiguity phenomena in natural language, and propose different computational models to tackle those problems. We show these models advance the state of the art. In our study of WSI, where sense categories are not explicitly defined but implicitly induced, we find that the state-of-the-art research in this field suffers from the ineffectiveness of the evaluation. We propose new evaluation approaches in Chapter 7, which,

we believe, will eventually lead to the advancement of WSI research. The detailed contributions of the thesis is as the follows:

Sense Granularity Line (different lexical ambiguity phenomena):

- We study the WSD problem and advance state-of-the-art.
- We propose computational models for a less well-studied problem, token-based idiom detection.
- We also study unlexicalized figurative expressions, and show that our methods can extract novel expressions from raw corpus automatically.
- Moving on to the more challenging problem of WSI, we propose alternative evaluation approaches, which, in our opinion, will eventually eliminate a major obstacle of this research topic.

Statistical Modeling Line:

- We undertake detailed feature engineering of a supervised model on the idiom detection task, which leads to improvements over state-of-the-art approaches.
- We propose a new bootstrapping model which reduces the human effort necessary for the supervised model, while maintaining a performance comparable to the supervised one.
- We propose a Gaussian Mixture Model which can be used to effectively discover new figurative expressions, an advancement of the work of idiom detection in the previous two chapters.
- We propose topic models which are based on Bayesian probabilistic theory to model WSD and idiom detection task within a uniform framework as long as sense paraphrases of different sense inventories are available. These models outperform state-of-the-art approaches.
- We show that the state-of-the-art WSI evaluation is not reliable. We propose alternative optimized strategies on the WSI evaluation based on the studies of entropy estimation simulation and stochastic prediction. Results show that our alternative strategies are favorable over state-of-the-art approaches.

1.4 Published Work

Part of the thesis has been published in previous conference papers: Chapter 3 is based on Li and Sporleder (2010a); Chapter 4 is based on Li and Sporleder (2009); Chapter 5 is based on Li and Sporleder (2010b); and Chapter 6 is based on Li et al. (2010). The work of Chapter 7 is done by joint supervision of Ivan Titov and Caroline Sporleder. It has been submitted to the journal Computational Linguistics under review.

2

Corpora and Resources

In this section, we introduce the corpora that we use for our experimental evaluation. We use four types of corpora: the Idiom Corpus (Chapter 3, 4 and 6), the Figurative Expression Corpus (Chapter 5), the Word Sense Disambiguation Corpora (Chapter 6) and the Word Sense Induction Corpora (Chapter 7).

The Idiom Corpus (UdSic) and the Figurative Expression Corpus (UdSfec) were constructed by ourselves. The Word Sense Disambiguation Corpora are from the SemEval 2007[1] shared task. We use datasets from two shared tasks: the Coarse-grained English All-Words Task (Navigli et al., 2007) for the coarse-grained word sense disambiguation evaluation and the English All-Words Task (Pradhan et al., 2007) for the fine-grained word sense disambiguation. The Word Sense Induction Corpora are from the SemEval 2007 and the SemEval 2010 [2] shared tasks: The first dataset is from the SemEval 2007 Evaluation of Word Sense Induction and Discrimination Systems (Agirre and Soroa, 2007); and the second dataset is from the SemEval 2010 Word Sense Induction and Disambiguation (Manandhar et al., 2010).

2.1 The Idiom Corpus: UdSic

We start by giving a detailed description of the UdSic corpus. The corpus contains 3964 records of 17 potential idiomatic expressions which were extracted from the Gigaword corpus (Table 2.1). All the records were annotated as idiomatic or non-idiomatic. The inter-annotator agreement on a small sample of doubly annotated examples was 97% and the kappa score

[1] See `http://nlp.cs.swarthmore.edu/semeval/`

[2] See `http://semeval2.fbk.eu/`

expression	literal	idiomatic	all
back the wrong horse	0	25	25
bite off more than one can chew	2	142	144
bite one's tongue	16	150	166
blow one's own trumpet	0	9	9
bounce off the wall*	39	7	46
break the ice	20	521	541
drop the ball*	688	215	903
get one's feet wet	17	140	157
pass the buck	7	255	262
play with fire	34	532	566
pull the trigger*	11	4	15
rock the boat	8	470	478
set in stone	9	272	281
spill the beans	3	172	175
sweep under the carpet	0	9	9
swim against the tide	1	125	126
tear one's hair out	7	54	61
all	862	3102	3964

Table 2.1: Idiom statistics (* indicates expressions for which the literal usage is more common than the idiomatic one)

0.7 (Cohen, 1960). For more details on this dataset, please refer to our previous papers (Li, 2008; Sporleder and Li, 2009). In this section, we introduce a follow-up of the previous corpus construction work. We construct a XML version of this corpus which integrates part-of-speech, lemma, dependency syntax, named entity and ID information. In the rest of this section, we show details of these corpus preprocessing work.

2.1.1 The Gigaword Corpus

The English Gigaword Corpus (Graff and Cieri, 2003) is produced and maintained by Linguistic Data Consortium (LDC). According to the LDC catalog, there are four distinct international English newswire collections in the corpus: Agence France Press English Service (afe), Associate Press Wordstream English Service (apw), the New York Times Newswire Service (nyt), and the Xinhua News Agency English Service (xie). This corpus covers a broad selection of topics, including politics, business, sports, entertainment among others.

2.1.2 Corpus Preprocessing

In our preprocessing step, the part-of-speech (POS) tag and lemmatization are done by RASP; the dependency parsing is done by MaltParser; and the named entity tagging is done by the Standford NE tagger. As a result of the incompatibility of the POS tags between RASP and MaltParser Trained Model, we apply MXPOST tagger to re-tag the corpus before using MaltParser.

RASP (Briscoe et al., 2006) is a parsing system for English. It is derived from potions of the ALvey NLP Tools[1]. It has multiple components such as a tokenizer (sentence boundary detection and tokenization), tagger (POS tagging), morphology analyzer (morphological analysis and generation) and parser (output the grammatical relations). RASP uses CLAWS C2 tagset (Jurafsky and Martin, 2000). We choose RASP because of the morphological analysis functionality, convenient for computing our lexical features. The lemmatization function outputs the lemma form, and the generation function generates different inflected forms of a lemma.

MXPOST tagger (Ratnaparkhi, 1996) is a POS tagger based on maximum entropy model. We use the Java implementation of the tagger published by the author. Unlike the RASP tagger, MXPOST uses the Penn Treebank part-of-speech tagset (Marcus et al., 1993), which allows us to tag our data in a format that can be accepted by the pre-trained MaltParser model.

MaltParser [2] (Nivre et al., 2006) is a statistical dependency parser. First it uses training data from a treebank to induce a model, and then utilizes this model to parse new input data. The parser itself is language independent. Furthermore, it is independent of the dependency tagset and the POS tagset. However, the pre-trained MaltParser model is only available in the Penn Treebank part-of-speech tagset (Marcus et al., 1993). The MaltParser developers converted the Penn Treebank data to dependency trees using the Standford Parser (Marneffe et al., 2006), so the pre-trained model outputs the Standford typed dependencies. Our preprocessed data uses the Standford typed dependencies.

[1] See `http://www.cl.cam.ac.uk/Research/NL/anlt.html`
[2] See `http://maltparser.org/index.html`

Stanford NER (Finkel et al., 2005) is a Java implementation of a Named Entity Tagger. This tagger adopts a Conditional Random Field (CRF) (Lafferty et al., 2001) sequence model. We use the version that contains three named entity classes (PERSON, ORGANIZATION, LOCATION) for English.

2.1.3 Data Format

We use a XML format, which consists of six nested elements, to store the preprocessed corpus.

- **"corpus" element** is the outermost element. The "id" property of this element records the name of the corpus.
- **"record" element** is nested under element "corpus", which has five distinct properties: "id" records the unique identity of the record within the corpus; "idiom" records the base form (dictionary form) of the idiom that this record contains (e.g., *break the ice*, *spill the beans*); "file" records the original Gigaword file that the record is extracted from (e.g., emphafe199407); "label" is the gold-standard human annotation which says whether the target expression is used literally or non-literally (e.g., *l* or *n*); "sid" records the id number of the record within the same idiom. (e.g., record containing *break the ice* with *sid="2"* means that it is the second record of the *break the ice* idiom set.)
- **"paragraph" element** is nested under element "record". The "id" property records the paragraph number as well as the record number. For instance, *id="2-3"* means that the current paragraph is the third paragraph of the second record.
- **"sentence" element** is nested under element "paragraph". Like the "id" property of the "paragraph" element, the "sentence id" records the position of the sentence. As an example, *id="2-3-1"* means that the current sentence is the first sentence of the third paragraph of the second record.
- **"word" element** is nested under "sentence" element. It contains seven properties: "id" records the position of the word within the sentence; "text" is the original word; "pos" is the part-of-speech tag by RASP; "lem" is the lemma of the word analyzed by RASP; "parent" is the id of dependency parent parsed by MaltParser; "deprel" is the dependency relation between this word and its parent; "ne" is the named entity tag of the word tagged by the Standford NER.

- **"iform" element** is nested under "word" element. It records different inflected forms of the word. In this corpus, we use the inflected forms of two types of words: Noun and Verb. For verbs, we record four types of inflected forms[1]: present participles (e.g., *wanting*), past tense (e.g., *wanted*), past participles (e.g., *wanted*) and third-person present-tense form (e.g., *wants*). For nouns, we record two type of inflected forms: singular (e.g., *box*) and plural (e.g., *boxes*).

2.2 Figurative Expression Corpus: UdSfec

In addition to the study of idiomatic expression, we also extend the topic to the study of general figurative (unlexicalized expressions) expressions. In this section, we describe the corpus used in figurative expression task described in Chapter 5.

2.2.1 Corpus Construction

We extract all the phrases with the POS pattern *V+det+N*[1] from the UdSic corpus (a subset of the Gigaword Corpus). We allow various inflected forms of verbs and nouns. Out of the total 7502 extractions, we randomly select a subset of 500 examples and label them manually as *literal* or *figurative*.

2.2.2 Corpus Annotation

To determine how well our model deals with different types of figurative usage, we distinguish four phenomena and define four labels (**nsa**, **nsu**, **nw**, **l**) for our annotation scheme.

Phrase level figurative means that the whole phrase is used figuratively. This category can be further divided into two subcategories: ambiguous figurative (**nsa**) and unambiguous figurative (**nsu**).

Ambiguous figurative (nsa) This category contains all the phrases that can have both 'literal' and 'nonliteral' readings. For instance, it is possible that an expression like "burning the bridge" takes on a literal meaning, while, a literal interpretation of the expression "trip the light fantastic"

[1]Those four types can cover the major varieties of the surface form of verbs. In our application, we focus on different surface forms of a same lemma.

[1]"V" is verb; "det" is determiner; "N" is noun.

does not make sense. Examples of this category extracted from the Gigaword Corpus are listed as the follows:

(2.1) If Green were making his comments about a team he used to work for, you would say he was burning his bridges behind him; what he's doing, instead, is almost like burning the bridge while he's still on it.

(2.2) In later years, I would come to understand that many successful revolutionaries enjoyed the fruits of success, and their status and were unwilling to give up either. Free elections would be permanently postponed and oppression equal to (or greater than) previous dictators would become part of the system.

(2.3) Dole showcased his legendary dry wit in a move that seemed designed to help soften the edges of steady attacks on the president he hopes to defeat in his third and final bid for the White house.

(2.4) Dr . Joy: Take the sock out of your mouth and create a brand-new relationship with your mom. Your mother is belittling you, literally trying to make you little, so she will still be the mom and you will still be the child. It's the only relationship she knows how to navigate with you. That's why it's up to you to change.

(2.5) Do not worry about Donald. You can go home and bet the ranch that he will be bigger and stronger than ever. Donald always wins.

Unambiguous figurative (nsu) This category contains expressions that can only have idiomatic reading. The literal reading violets grammatical rules, selectional constraints or common knowledge. (See Examples 2.6 and 2.7)

(2.6) McGwire's answer to Sosa has been swift. Is it this easy for Mcgwire? Some, including Sosa, have suggested that McGwire has the edge in controlling the pace of this race because the cardinals are out of the playoffs. No pressure, no holding back. Mcgwire can relax and swing freely.

(2.7) Trip the light fantastic is an extravagant way of referring to dancing, a phrase rather more common years ago than it is now.

Token level figurative (nw) is also called Weak Figurative. The label *token-level figurative* (**nw**) is used when part of the phrase is used figuratively (e.g., *sparrow* in (2.8)). Often it is difficult to determine whether a word is still used in a 'literal' sense (e.g., lexicalized in a dictionry) or whether it is used figuratively. Since we are interested in improving the performance of NLP applications such as MT, we take a pragmatic approach and classify usages as 'figurative' if they are not lexicalized, i.e., if the specific sense is not listed in a dictionary.[1] For example, we would classify *summit* in the *meeting* sense as 'literal' (**l**), and for the same reason, we treat *steer* in Example 2.9 and *jack* in Example 2.10 as literal.

(2.8) During the Iraq war, he was a sparrow; he didn't condone the bloodshed but wasn't bothered enough to go out and protest.

Literal (l) All the cases which are clearly of literal usage. As discussed in the **nw** case, we also annotated sense lexicalized expressions as literal (e.g., "steer the industry", "jack the price").

(2.9) But the main point is that by taking leadership Intel has filled a serious void. IBM, of course, originally defined the PC standard, but during the course of the 1980s, as it lost market share to what were then called IBM compatibles, or clones. It lost the clout to steer the industry in new directions. Compaq tried, notably with its Eisa card standard, but few of its jealous rivals were willing to follow.

(2.10) The cartel would love to jack the price higher; That's what cartels do, by managing supply to jigger up demand. But some oil-producing country always finds the temptation too great, and begins to cheat on production quotas, flooding the market, driving prices down. I mean, would you like to be in a position where you had to trust an oil-producing nation not to get grabby?

In summary, *Phrase-level figurative* means that the whole phrase is used figuratively. We further divide this class into expressions which are potentially ambiguous between literal and figurative usages (**nsa**), e.g., *spill the beans*, and those that are unambiguously figurative irrespective of the context (**nsu**), e.g., *trip the light fantastic*. The latter can, theoretically, be detected by dictionary look-up, while the former have to be detected in context. The label *token-level figurative* (**nw**) is used when part of the phrase is used figuratively.

[1] We use `http://www.askoxford.com`.

2.2.3 Discussion and Statistics

In our annotation, we noticed that some target expressions are actually named entities (e.g., movie name, event name, music or band name, etc., see Example 2.11 and 2.12). "Inherit the Wind" is actually a name of the movie. We also find that there are a certain number of conventional domain specific phrases: In Example 2.13, the target expression "loading the bases" is a technical term in the sport of baseball.[1] We treat both named entities and domain conventional phrases as 'figurative' in our practice.

(2.11) MGM salted Kelly's song-and-dance movie career with a few dramatic roles, "Living in a Big Way" (1947), "The Three Musketeer's" (1948), "It's a Big Country" (1951) and "Inherit the Wind" (1960).

(2.12) Part of the problem for Lockheed Martin, which two years ago won a fierce competition for a $900 million NASA contract to build an experimental reusable rocket called the **x-33**, may be that the company simply bit off more than it could chew, with a project that would ultimately require it to assume all the costs of commercial development even though many observers think the company has little incentive or commitment to do so.

(2.13) Pettitte battled through the middle innings before finally crumbling in the sixth, walking Alex Ochoa to lead off. Pat Meares and Ortiz singled, loading the bases. Pettitte got a force play at home and a short fly out to right, and it seemed he would escape the inning.

In all, we annotated 500 records in total, of which, 7.3% of the instances were annotated as 'nsa', 1.9% as 'nsu', 9.2% as 'nw' and 81.5% as 'l'. A randomly selected sample set (100 instances) was annotated independently by a second annotator. The kappa score (Cohen, 1960) is 0.84, which suggest that the annotations are reliable.

2.3 Word Sense Disambiguation Corpora

As introduced in the introduction chapter, our studies also cover various WSD phenomena. The corpora described in this section are used for the WSD task Chapter 6. We introduce two WSD corpora: one is for the fine-grained WSD; and the other is for the coarse-grained WSD.

[1] In baseball, the phrase "loading the bases" refer to the event of causing the bases to become loaded.

2.3.1 SemEval 2007 Fine-grained WSD Dataset

The fine-grained WSD dataset is provided by Pradhan et al. (2007) for the Semeval 2007 Task-17 (English Fine-grained All-words Task). This dataset is a subset of the set from Task-07. It comprises the three WSJ articles from Navigli et al. (2007). A total of 465 lemmas were selected as instances from about 3500 words of text. There are 10 instances marked as 'U' (undecided sense tag). Of the remaining 455 instances, 159 are nouns and 296 are verbs. The sense inventory is from WordNet 2.1.

The organizers do not supply the part-of-speech and lemma information of the target instances. In order to avoid the wrong predictions caused by tagging or lemmatization errors, we manually corrected any bad tags and lemmas for the target instances. This was done by comparing the predicted sense keys and the gold standard sense keys. We only checked instances for which the POS-tags in the predicted sense keys are not consistent with those in the gold standard. This is the case for around 20 instances.

2.3.2 SemEval 2007 Coarse-grained WSD Dataset

The coarse-grained WSD dataset is from the Semeval 2007 Task-07 benchmark dataset released by Navigli et al. (2007). As reported in the paper, the dataset consists of 5377 words of running text from five different articles: the first three were obtained from the WSJ corpus, the fourth was the Wikipedia entry for *computer programming*, and the fifth was an excerpt of Amy Steedman's *Knights of the Art*, biographies of Italian painters. The proportion of the non news text, the last two articles, constitutes 51.87% of the whole testing set. In total, this corpus consists of 1108 nouns, 591 verbs, 362 adjectives, and 208 adverbs. The data were annotated with coarse-grained senses which were obtained by clustering senses from the WordNet 2.1 sense inventory based on the procedure proposed by Navigli (2006).

2.4 Word Sense Induction Corpora

We also conduct research on Word Sense Induction which is a subsequent work of Word Sense Disambiguation. As described in the introduction section, WSI faces with evaluations which hinders the research in this field. In this section, we describe two datasets from the SemEval shared tasks that we use for the study of the evaluation problems of WSI.

2.4.1 SemEval 2007 WSI Dataset

The SemEval 2007 WSI dataset is from task-02 "Evaluation Word Sense Induction and Discrimination Systems" (Agirre and Soroa, 2007). It is borrowed from the SemEval 2007 "English lexical sample subtask" of task-17 (Pradhan et al., 2007). The texts are taken from the Wall Street Journal and Brown corpora. All the instances are human annotated with OntoNotes senses (Hovy et al., 2006). There are 100 target words in this dataset (65 verbs and 35 nouns). The sense tags are removed from the training corpus. All the training and testing examples are combined together. This results a total number of 27,132 instances (17,649 nouns and 12,200 verbs).

2.4.2 SemEval 2010 WSI Dataset

The second WSI dataset is from the SemEval 2010 "Word Sense Induction" task (Manandhar and Klapaftis, 2009). All the instances of the dataset are taken from OntoNotes. Each instance consists of a maximum of three sentences. The source of those texts are mainly newswires such as the Wall Street Journal, CNN and BBC. There are 100 target words in this dataset (50 nouns and 50 verbs). The organizers supply a test set containing 8,915 manually annotated examples. In the gold annotation, the average number of sense clusters is 5.6.

2.5 Wikipedia Dump

In addition to the evaluation datasets described in the last few sections, we also use a Wikipedia Dump (Roth and Klakow, 2010)[1] for our topic estimation experiments (Chapter 6). This dataset, which consists of 320,000 articles,[2] is significantly larger than other standard NLP corpora (e.g., BNC or Gigaword). All markup from the Wikipedia dump was stripped off using the same filter as the ESA implementation (Sorg and Cimiano, 2008), and stopwords were filtered out using the Snowball (Porter, October 2001) stopword list. In addition, words with a Wikipedia document frequency of one were filtered out. The lemmatized version of the corpus consists of 299,825 lexical units.

[1] The version is from the English snapshot of 2009-07-13.

[2] All articles of fewer than 100 words were discarded.

3

A Supervised Model to Disambiguate Idiomatic Expressions

We start with the problem *token-based idiom detection* (given a potentially idiomatic phrase in context, decide whether it is used literally or idiomatically), in which the lexicalized phrases (dictionary form) have two possible senses 'literal' or 'idiomatic'. In this chapter, we investigate supervised models for this task. We are specifically interested in which types of features (e.g., semantic relatedness features, local context, global context, syntactic properties and other linguistic indicators) perform best and more specifically which features generalize across idioms. We compare the results with state-of-the-art together for this task.

3.1 Introduction

Nonliteral expressions are a major challenge in NLP because they are (i) fairly frequent and (ii) often behave idiosyncratically. Apart from typically being semantically more or less opaque, they can also disobey grammatical constraints (e.g., *by and large*, *lie in wait*). Hence, idiomatic expressions are not only a problem for semantic analysis but can also have a negative effect on other NLP applications (Sag et al., 2001), such as parsing (Baldwin et al., 2004).

To process nonliteral language correctly, NLP systems need to recognize such expressions automatically. While there has been a significant body of work on idiom (and more generally multi-word expression) detection (see Section 3.5), until recently most approaches have focused on a *type-based classification*, dividing expressions into "idiomatic" or "not idiomatic" irrespective of their actual use in a discourse context. However, while some expressions, such as

by and large, always have a non-compositional, idiomatic meaning, many other expressions, such as *break the ice* or *spill the beans*, can be used literally as well as idiomatically and for some expressions, such as *drop the ball*, the literal usage can even dominate in some domains (e.g., *sports*). Consequently, those expressions have to be disambiguated in context (*token-based classification*). See Examples 3.1 and 3.2 for concrete examples of idiomatic phrases being used literally.

(3.1) Dad had to break the ice on the chicken troughs so that they could get water.

(3.2) Somehow I always end up spilling the beans all over the floor and looking foolish when the clerk comes to sweep them up.

In this chapter, we investigate how well models for distinguishing literal and nonliteral use can be learned from annotated examples. We explore different types of features, such as the local and global context, syntactic properties of the local context, the form of the expression itself and properties relating to the semantic relatedness structure of the discourse. We show that several feature types work well for this task. However, some features can generalize across specific idioms, for instance features which compute how well an idiom "fits" its surrounding context under a literal or nonliteral interpretation. This property is an advantage because such features are not restricted to training data for a specific target expression but can also benefit from data for other idioms. This is important because, while idioms as a general linguistic class are relatively frequent, instances of each particular idiom are much more difficult to find in sufficient numbers. The situation is exacerbated by the fact the distributions of literal vs. nonliteral usage tend to be highly skewed, with one usage (often the nonliteral one) being much more frequent than the other. Finding sufficient examples of the minority class can then be difficult, even if instances are extracted from large corpora.

We show that it is possible to circumvent this problem by employing a generic feature space that looks at the cohesive ties between the potential idiom and its surrounding discourse. Such features generalize well across different expressions and lead to acceptable performance even on expressions unseen in the training set.

3.2 Modeling Semantic Relatedness

In this section, we introduce how we model semantic relatedness, which is the basis of the semantic relatedness induced features (relW, relS and connect., described in Section 3.3). As

modeling semantic relatedness is a very active research area in computational linguistics, various similarity measures have bee proposed in previous studies (Chen et al., 2006; Herdağdelen et al., 2009; Pedersen et al., 2004; Rubenstein and Goodenough, 1965). We chose a measure called *Normalized Google Distance* (NGD) Cilibrasi and Vitanyi (2007), which computes relatedness on the basis of page counts returned by a search engine.[1] It is defined as follows:

$$NGD(x,y) = \frac{max\{log\ f(x), log\ f(y)\} - log\ f(x,y)}{log\ M - min\{log\ f(x), log\ f(y)\}} \tag{3.3}$$

where x and y are the two words whose association strength is computed (e.g., *fire* and *coal*), $f(x)$ is the page count returned by the search engine for x (and likewise for $f(y)$ and y), $f(x,y)$ is the page count returned when querying for "x AND y", (i.e., the number of pages that contain both, x and y), and M is the number of web pages indexed by the search engine. The basic idea is that the more often two terms occur together, relative to their overall occurrence, the more closely they are related.

In two previous experiments (Li, 2008), NGD is shown to be highly correlated to semantic relatedness rated by humans on the German data set from *Technische Universität Darmstadt* [2] and on the English dataset *the WordSimilarity-353 Test Collection* [3]. In the next section, we describe a new experiment in which we compare the NGD value with 'literal'/'nonliteral' lexical chains identified by human annotators. We find that the NGD value generally agrees well with human intuition, which further justifies our decision to use NGD as a semantic relatedness measure for extracting semantic relatedness based features for our supervised model.

3.2.1 Comparing NGD with Human Annotation

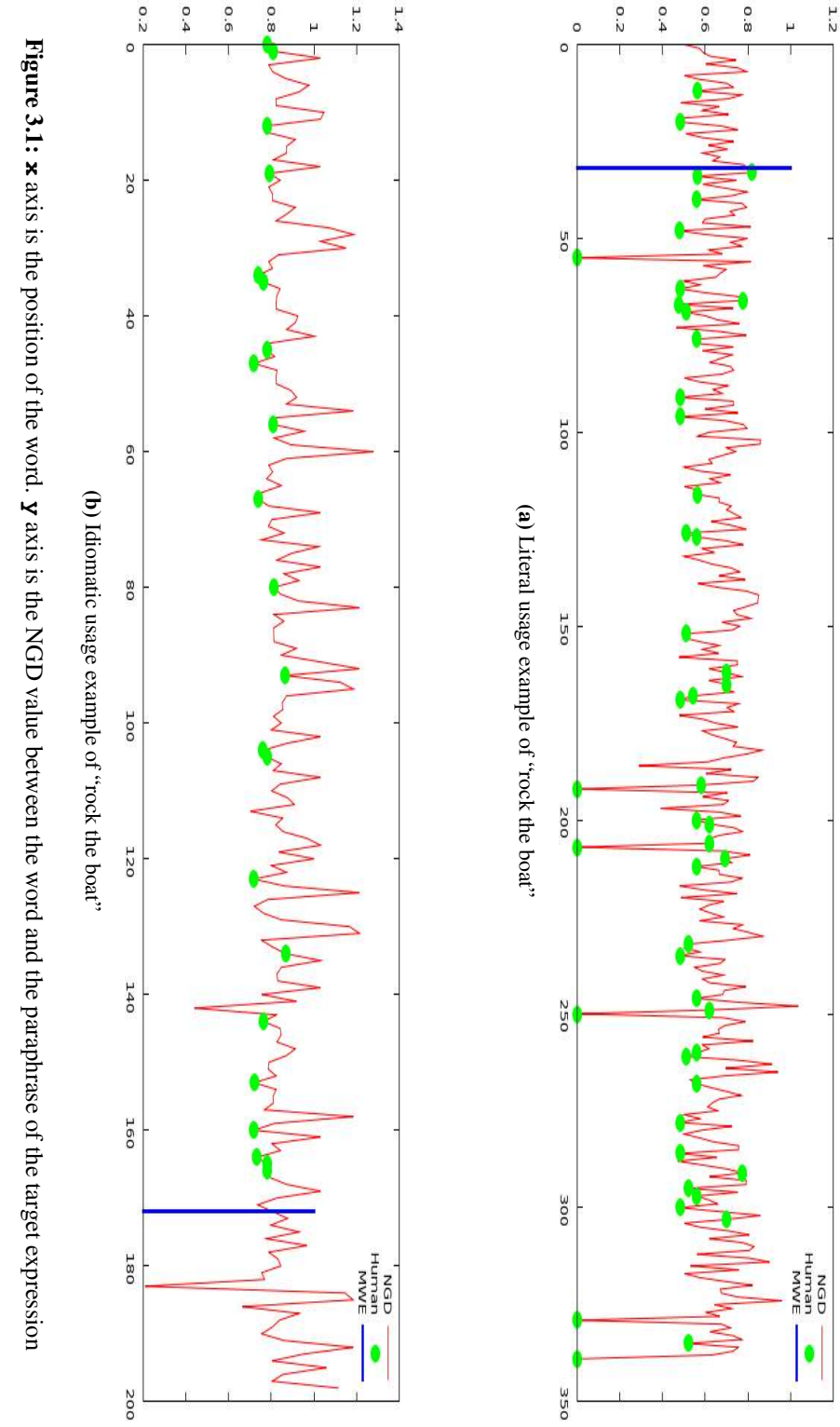
We use the annotated dataset from Sporleder et al. (to appear), in which two annotators annotated the lexical chain words in texts that indicate the 'literal' or 'idiomatic' use of potential idiomatic expressions.[4] We compare NGD value with the human annotation to check whether human judged chain words correlate with a high semantic relatedness value predicted by our automatic semantic relatedness modeling approach (low NGD).

[1] We employ Yahoo! rather than Google since we found that it returns more stable counts.

[2] Available from `http://www.ukp.tu-darmstadt.de/data/semantic-relatedness/`; All subjects in the experiments were native speaker of German, they were asked to rate the word pairs by similarity on a scale of 0-4.

[3] Available from the computer science department of *Technion - Israel Institute of Technology:* `http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/`

[4] The annotation was adjudicated by an American English native speaker.



(a) Literal usage example of "rock the boat"

(b) Idiomatic usage example of "rock the boat"

Figure 3.1: x axis is the position of the word. **y** axis is the NGD value between the word and the paraphrase of the target expression

In our automatic approach, the semantic relatedness of the literal chain words are modeled by the NGD between the word and the component words in the expression (e.g., *boat* for 'literal' *rock the boat*), while the relatedness of the nonliteral chain words is modeled by the NGD between the word and the idiomatic paraphrase (idiom definition. e.g., the paraphrase of idiomatic *rock the boat* is *upset conventions, break norms, cause trouble*.). The relatedness of a literal chain word w_l with the literal phrase *rock the boat*, for instance, would be $NGD(w_l, `boat')$, whereas the relatedness of a nonliteral chain word $w_{non}$ with the nonliteral phrase *rock the boat* would be the $NGD(w_{non}, `upset\ conventions\ OR\ break\ norms\ OR\ cause\ trouble')$.[1]

As examples, Figures 3.1a and 3.1b plot the NGD for a given word against its position in the text. This allows us to see how different words are related to the meaning of the target expression within the context. The position of the target expression in the text is marked by a (blue) vertical line. Words that are marked as semantically related by human annotators are indicated by a (green) circle. Figure 3.1a shows the results for a literal example of *rock the boat*, while Figure 3.1b shows the results for a nonliteral example of the same idiom. The original texts of the plots are in the appendix (see Appendix **??** for the literal example, and Appendix **??** for the idiomatic example).

We find that human annotations agree quite well with the NGD values, as words marked by humans tend to be located at local minimal in the graph. NGD is able to identify semantically related words marked by humans. This general pattern is observable for both the idiomatic and the literal meaning. Therefore, we demonstrate that modeling semantic relatedness by NGD is an effective strategy.

We also find that the NGD value of the associated words (marked by humans) in the literal examples is lower than in the nonliteral examples. While most of the associated words in the literal cases have an NGD value of around 0.5, most of the words in the nonliteral cases have an NGD value of around 0.8 (see Figure 3.1). Our further study suggests that idiomatic readings tend to appear in rather diverse contexts. For instance, *rock the boat* can mean *cause trouble* or *upset conventions*. It is more likely that words such as *accusation, attack, conflict* co-occur with the first reading, while words such as *counterculture, rebels, change, norm* co-occur with second reading. The diversity of nuances to the idiomatic meaning leads to a scattered distribution of the idiomatic meaning across many different context words. As a result, the nonliteral NGD is generally high (i.e., words tend to be rated as not very similar to the idiomatic meaning). It

[1] Computationally, we represent the paraphrases by using the OR logic operator to connect all the possible paraphrases when sending a query to the search engine.

actually closely resembles human intuition, as humans also rate semantic related links with idiomatic meanings as relatively weak. A further supporting argument is that results also show that human judges tend to annotate more words in the literal example than the nonliteral example (see Figure 3.1a and Figure 3.1b). We also find that there is also more disagreement in the nonliteral annotation compared with literal annotation. In all, our study suggests that the nonliteral semantic relatedness is not only more difficult to identify for the automatic method but also for human judges. As a result, we decide only include literal semantic related features in our supervised model as nonliteral semantic related features are more difficult to capture and including them is more likely to introduce noise to later stage procedures.

3.3 Features of Idiomatic and Literal Usage

In this study we are particularly interested in which features work well for the task of distinguishing literal and idiomatic language uses. The few previous studies have mainly looked at the lexical context in which and expression occurs (Birke and Sarkar, 2006; Katz and Giesbrecht, 2006). However, other properties of the syntactic and semantic features might also be useful. We distinguish these features into different groups and discuss them in the following sections.

3.3.1 Discourse Cohesion (dc)

We start this section by a brief description of the cohesion-graph approach proposed by Li (2008); Sporleder and Li (2009), on which our Discourse Cohesion feature extraction is build up. This model exploits the fact that words in a coherent discourse exhibit *lexical cohesion* (Halliday and Hasan, 1976), i.e. concepts referred to in sentences are typically related to other concepts mentioned elsewhere in the discourse. Given a suitable measure of semantic relatedness, it is possible to compute the strength of such cohesive ties between pairs of words. While the component words of literally used expressions tend to exhibit lexical cohesion with their context, the words of nonliterally used expressions do not. For example, in (3.4) the expression *play with fire* is used literally and the word *fire* is related to surrounding words like *grilling*, *dry-heat*, *cooking*, and *coals*. In (3.5), however *play with fire* is used nonliterally and cohesive ties between *play* or *fire* and the context are absent.

(3.4) **Grilling** outdoors is much more than just another **dry-heat cooking** method. It's the chance to play with **fire**, satisfying a primal urge to stir around in **coals** .

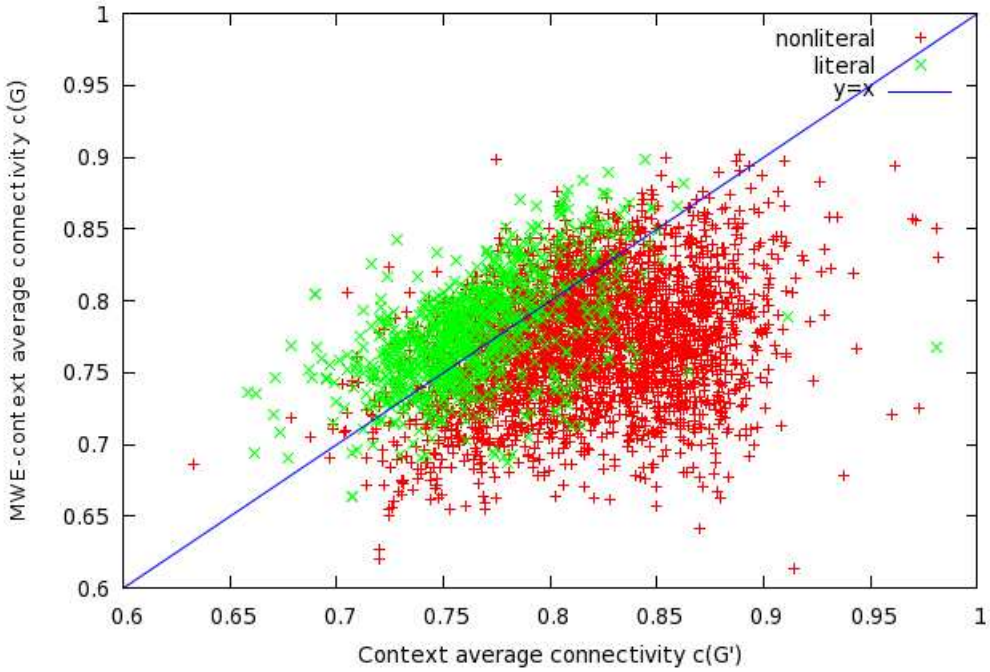


Figure 3.2: Idiom instances represented in the discourse connectivity feature space, $y = x$ is decision boundary by the cohesion graph, $c(G')$ is the average connectivity of the discourse, $c(G)$ is the average connectivity between the idiom component words and the context.

(3.5) And PLO chairman Yasser Arafat has accused Israel of playing with fire by supporting HAMAS in its infancy.

To determine the strength of cohesive links, the unsupervised model builds a graph structure (called *cohesion graph*) in which all pairs of content words in the context are connected by an edge which is weighted by the pair's semantic relatedness. Then the *connectivity* of the graph is computed, defined as the average edge weight. If the connectivity increases when the component words of the idiom are removed, then there are no strong cohesive ties between the expression and the context and the example is labelled as 'nonliteral', otherwise it is labelled as 'literal'.

We implement two semantic relatedness discourse cohesion features, *discourse connectivity* and *related score* , which take into account the cohesive structure of an expression in context. These features look at the lexical semantic relatedness between an expression and the surrounding context, so they are more likely to generalize across different idioms.

Discourse Connectivity (connect.) In the cohesion graph approach, the decision boundary is defined by:

$$\Delta c = c(G) - c(G^{'}) \tag{3.6}$$

where, $c(G)$ is the average connectivity between the idiom component words and the context; $c(G^{'})$ is the average connectivity of the discourse. The idiom component words are likely be related to the context in the literal cases, so $\Delta c > 0$ can be used to identify literal examples. We conduct an experiment to visualize this process, in which all the idiom data instance are represented by the two discourse connectivity features (see Figure 3.2). The decision boundary of the graph classifier can be represented as the line $x = y$. However, a further research step can take these two values of connectivity as features for a supervised learning model and find more sophisticated classification boundaries (e.g., nonlinear boundary).

Based on this idea, we implement two features which look at the cohesion graph of an instance. We encode the connectivity of the graph (i) when the target expression is included and (ii) when it is excluded. The cohesion graph classifier uses the difference between these two values to make its prediction. By encoding the absolute connectivity values as features we enable the supervised classifier to make use of this information to model more complicated decision boundaries.

Relatedness Score (relS) The feature set implements the *relatedness score* which encodes the scores for the 100 most highly weighted edges in the cohesion graph (ranked by NGD).[1] If these scores are high, there are cohesive links with the discourse and the target expression is likely to be used literally. For instance, the top one related score feature in the literal case (Example 3.7) is the related score between *ice* and *water*. In contrast, the top one related score in the idiomatic case (Example 3.8) is the related score between *ice* and *games*. The related score is higher in the literal case than in the idiomatic case.

(3.7) The **water** would break the ice surface with its ccumulated energy.

(3.8) We played a couple of party **games** to break the ice.

[1] We only used the 100 highest ranked edges because we are looking at a specific context here rather than the contexts of the literal or nonliteral class overall. Since the contexts we use are only five paragraphs long, recording the 100 strongest edges seems sufficient.

3.3.2 Global Lexical Context (glc)

That intuition that lexical context may be a good indicator for the usage of an expression is indicated by examples such as (4.1) and (3.10), which suggest that literal and nonliteral usages of a specific idiom co-occur with different sets of words. For instance, nonliteral uses of *break the ice* (4.1) tend to occur with words like *discuss*, *bilateral* or *relations*, while literal usages (3.10) predictably occur with, among others, *frozen*, *cold* or *water*. What we look at here is the global lexical context of an expression, i.e., taking into account previous and following sentences. We specifically look for words which are either correlated (in a wide sense) to the literal or the nonliteral sense of the target expression. The presence or absence of such words is indicator of how the expression is used in a context.

(3.9) "Gujral will meet Sharif on Monday and **discuss bilateral relations**," the Press Trust of India added. The minister said Sharif and Gujral would be able to "break the ice" over Kashmir.

(3.10) Meanwhile in Germany, the **cold** penetrated Cologne cathedral, where worshippers had to break the ice on the **frozen** holy **water** in the font.

In all, we implement two sets of features which encode the global lexical context: *salient words* and *related words*. The former feature uses a variant of tf.idf to identify words that are particularly salient for 'literal' or 'idiomatic' usages. The latter feature identifies words which are most strongly semantically related to the component words of the target expression.

Salient Words (salW) This feature aims to identify words which are particularly *salient* for literal usage. We use a frequency-based definition of salience and compute the *literal saliency score* for each word in a five-paragraph context around the target expression:

$$sal_{lit}(w) = \frac{i_{lit}(w) \times \log f_{lit}(w)}{i_{nonlit}(w) \times \log f_{nonlit}(w)} \tag{3.11}$$

where $sal_{lit}(w)$ is the saliency score of the word w for the class lit; $f_{lit}(w)$ is the token frequency of the word w for literally used expressions; $i_{lit}(w)$ is inverse number of instances

of the target expressions classified as lit which co-occur with word w (and mutatis mutandis $nonlit$ for target expressions labelled as nonliteral).[1]

Words with a high sal_{lit} occur much more frequently with literal usages than with nonliteral ones. Conversely, words with a low sal_{lit} should be more indicative of the nonliteral class. However, we found that, in practice, the measure is better at picking out indicative words for the literal class; nonliteral usages tend to co-occur with a wide range of words. For example, among the highest scoring words for *break the ice* we find *thick, bucket, cold, water, reservoir* etc. While we do find words like *relations, diplomacy, discussions* among the lowest scoring terms (i.e., terms indicative of the nonliteral class), we also find a lot of noise (*ask, month*). The effect is even more pronounced for other expressions (like *drop the ball*) which tend to be used idiomatically in a wider variety of situations (*drop the ball on a ban of chemical weapons*, *drop the ball on debt reduction* etc.).

We implement the saliency score in our model by encoding for the 300 highest scoring words whether the word is present in the context of a given example and how frequently it occurs.[1] Note that this feature (as well as the next one) can be computed in a per-idiom or a generic fashion. In the former case, we would encode the top 300 words separately for each idiom in the training set, in the latter across all idioms (with the consequence that more frequent idioms in the training set contribute to more positions in the feature vector).

Furthermore, we also find there are cases in which several idioms co-occur within the same instance, as the writers convey a stylish writing style (e.g., irony) by excessively usage of idioms. Consequently, global lexical context features may also generalize across idioms to some extend.

Related Words (relW) This feature set is a variant of the previous one. Here we score the words not based on their saliency but we determine the semantic relatedness between the noun in the idiomatic expression and each word in the global context, using the *Normalized Google Distance* mentioned in Section 3.2. We encode the 300 top-scoring words. While the *related words* feature is less prone to overestimation of accidental co-occurrence than the saliency

[1]Our definition of sal_{lit} bears similarities with the well known $tf.idf$ score. We include both the term frequencies (f_{lit}) and the instance frequencies (i_{lit}) in the formula because we believe both are important. However, the instance frequency is more informative and less sensitive to noise because it indicates that expression classified as 'literal' *consistently* co-occurs with the word in question. Therefore we weight down the effect of the term frequency by taking its log.

[1]We also experimented with different feature dimensions besides 300 but did not find a big difference in performance.

feature, it has the disadvantage of conflating different word senses. For example, among the highest scoring words for *ice* are *cold, melt, snow, skate, hockey* but also *cream, vanilla, dessert*.

3.3.3 Local Lexical Context (locCont)

In addition to the global context, the local lexical context, i.e., the words preceding and following the target expression, also provide important information. Some very frequent local clues are words such as *literally* or *metaphorically speaking*. Unfortunately, such clues are not only very rare (we only found a handful in nearly 4,000 annotated examples) but also not always reliable. For instance, it is not difficult to find examples like (3.12) and (3.13) where the word *literally* is used even though the idiom clearly has a nonliteral meaning.

(3.12) In the documentary the producer **literally** spills the beans on the real deal behind the movie production.

(3.13) The new philosophy is blatantly permissive and **literally** passes the buck to the House's other committees.

We also find more local cues examples on idiom specific level. For example, the word *just* before *get ones feet wet* often indicates nonliteral (see Example 3.14). As another example, the occurrence of the prepositions *over* or *between* after *break the ice* often indicate 'nonliteral' (see Example 4.1 and 3.15). Although such cues are not perfect they often make one usage more likely than the other. Unlike the semantically based global cues, many local clues are more rooted in syntax, i.e., local cues work because specific *constructions* tend to be more frequent for one class than the other.

(3.14) The wiki includes a page of tasks suitable for those **just** getting their feet wet.

(3.15) Would the visit of the minister help break the ice **between** India and Pakistan?

Another type of local cues involve selectional preferences. For instance, idiomatic class is very likely if the subject of the verb phrase *play with fire* is a country (Example 3.16) or if the phrase *break the ice* is followed by a *with*-PP whose NP refers to a person (Example 3.17).

(3.16) Dudayev repeated his frequent warnings that **Russia** was playing with fire.

(3.17) Edwards usually manages to break the ice with the taciturn **monarch**.

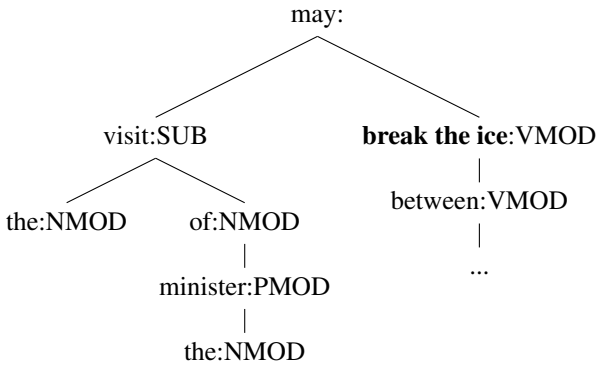


Figure 3.3: Dependency tree for a nonliteral example of *break the ice* (*The visit of the minister may break the ice between India and Pakistan.*)

Based on those observations, we encode words occurring in a ten word window around the target expression, five pre-target words and five post-target words, as the locCont features.

3.3.4 Syntactic Structure (allSyn)

To encode syntactic features, we first select the *head node (heaSyn)* of the target expression from the dependency tree (e.g., *break* in Figure 3.3). Then, we select further features by the *parent node (parSyn)*, the *sibling nodes (sibSyn)* and the *children nodes (chiSyn)* of the *head node (heaSyn)*. Altogether, these nodes include the following categories of syntactic information:

Dependency Relation of the Verb Phrase The whole idiomatic expression used as an object of a preposition is indicative of idiomatic usage (see Example 3.18). This property is captured by the *heaSyn* node.

(3.18) Ross headed back last week to Washington to brief president Bill Clinton on the Hebron talks after achieving a breakthrough **in** breaking the ice in the Hebron talks by arranging an Arafat-Netanyahu summit .

Modal Verbs usually appear in the parent position of the head verb (*parSyn*). Modals can be an indicator of idiomatic usage such as *may* in Figure 3.3. In contrast, the modal *had to* is an indicator that *break the ice* is used literally (Figure 3.4).

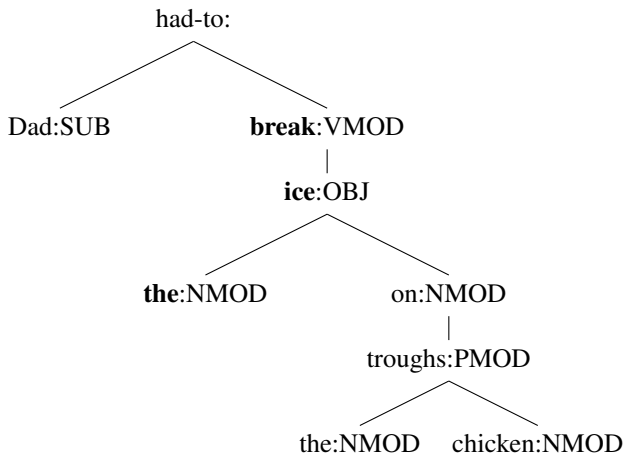


Figure 3.4: Dependency tree for a literal example of *break the ice* (*Dad had to break the ice on the chicken troughs.*)

Subjects also provide clues about the usage of an expression (e.g., disobey selectional preferences). For instance, *visit* as a subject of the verb phrase *break the ice* is an indicator of idiomatic usage (see Figure 3.3). As another example, subjects typically appear in the children position of the head verb, while, in unconventionally cases (idiomatic), they may appear in the sibling position (as Figure 3.3).

Verb Subcat We further encode the arguments of the head verb of the target expression. These arguments can be, for example, additional PPs. This feature encodes syntactic constraints and aims to model selectional restrictions. The subcategorisation frames often differ from each other in the two cases, e.g., nonliteral expressions often tend to have shorter argument lists than literal ones. For instance, the subcat frame <PP-on, PP-for> intuitively seems more likely for literal usages of the expression *drop the ball* (Example 3.19) than for nonliteral ones, for which <PP-on> is more likely (Example 3.20). In our experiments, the children nodes of the head node (*chiSyn*) encode the subcategorisation frames.

(3.19) US defender Alexi Lalas twice went close to forcing an equaliser , first with a glancing equaliser from a Paul Caligiuri free kick and then from a Wynalda corner when Prunea dropped the ball **[on the ground]** only **[for Tibor Selyme to kick frantically clear]**.

(3.20) "Clinton dropped the ball **[on this]**," said John Parachini.

Modifiers of the verb are also indicative of the 'literal' or 'idiomatic' usage. Take 3.21 as an example, the fact that the phrase *get one's feet wet* is modified by the adverb *just* suggests a idiomatic use. Similar to the verb subcat, modifiers are often appear in the children node position in the dependency tree (*chiSyn*).

(3.21) The wiki includes a page of tasks suitable for those **just** getting their feet wet.

Coordinated Verb Which verbs are coordinated with the target expression, if any, also provides cues for the intended interpretation. For example, in (3.22), the fact that *break the ice* is coordinated with another verb *fall* suggests that the phrase is used literally. The coordinated verb may appear at the sibling, children, or another position of the head verb depending on the dependency parser. We use MaltParser, which tends to output the coordinated verbs in the children position of the first verb.

(3.22) They may break the ice and **fall** through.

3.3.5 Other Features

Named Entities (ne) Diab and Bhutada (2009) find that NE-features are useful for idiom detection task and they use a commercial NE-tagger with 19 classes in their experiments. We also find NEs are indicative by out preliminary data study. For instance (3.16), a country name as subject of the phrase *break the ice* often indicates idiomatic usage. We use the Stanford NE tagger (Finkel et al., 2005), and encode three named entity classes ("person", "location", "organization") as NE features.

Indicative Terms (iTerm) Words like *literally, proverbially* are often indicative of literal or idiomatic usages as well. We encode the frequencies of such terms as iTerm features.

Scare Quotes (quote) feature encodes whether the idiom is marked off by scare quotes, as it often indicates nonliteral usages (3.23).

(3.23) Do consider "getting your feet wet" online, using some of the technology that is now available to us.

3.4 Experiments

In the previous section we discuss different features for idiom disambiguation task. To determine which of these features work best for the task and which ones generalize across different idioms, we carry out three experiments. In the first one (Section 3.4.1) we train one model for each idiom (see Chapter 2) and test the effectiveness of each feature type individually as well as different feature combinations. In the second experiment (Section 3.4.2), we train one generic model for all idioms and determine how the performance of this model differs from the idiom-specific models. Specifically we want to know whether the model would benefit from the additional training data available by combining information from several idioms. Finally (Section 3.4.3), we test the generic model on *unseen* idioms to determine whether these can be classified based on generic properties even if training data for the target expressions have not been seen.

3.4.1 Idiom Specific Models

The first question we want to answer is how difficult token-based idiom classification is and which of the features defined in the previous section work well for this task. We implement a specific classifier for each of the idioms in the data set. We train models for different feature combinations and for each individual feature. Because the data set is not very big we decide to run these experiments in 10-fold stratified cross-validation mode. We use the SVM classifier (SMO) from Weka.[1]

Table 3.1 shows the results. We report the precision (Prec.), recall (Rec.) and F-Score for the literal class, as well as the accuracy. Note that due to the imbalance in the data set, accuracy is not a very informative measure here; a classifier always predicting the majority class would already obtain a relatively high accuracy. The literal F-Score obtained for individual idioms varies from 38.10% for *bite one's tongue* to 96.10% for *bounce of the wall*. However, the data sets for the different idioms are relatively small and it is impossible to say whether performance differences on individual idioms are accidental, or due to differences in training set size or due to some inherent difficulty of the individual idiom. Thus we chose not to report the performance of our models on individual idioms but on the whole data set for which the numbers are much more reliable. The final performance confusion matrix is the sum over all individual idiom confusion matrices.

[1] `http://www.cs.waikato.ac.nz/ml/weka/`

	Avg. literal			Avg.
feature	Prec.	Rec.	F-Score	Acc.
all	89.84	77.06	82.96	93.36
glc+dc	90.42	76.44	82.85	93.36
allSyn	76.30	86.13	80.92	91.48
heaSyn	76.64	85.77	80.95	91.53
parSyn	76.43	88.34	81.96	91.84
chiSyn	76.49	88.22	81.94	91.84
sibSyn	76.27	88.34	81.86	91.78
locCont	76.51	88.34	82.00	91.86
ne	76.49	88.22	81.94	91.84
iTerm	76.51	88.34	82.00	91.86
quote	76.51	88.34	82.00	91.86
Base_{maj}	76.71	88.34	82.00	91.86

Table 3.1: Performance of idiom-specific models (averaged over different idioms), 10-fold stratified cross-validation.

The Baseline (Base_{maj}) is built based on predicting the majority class for each expression. This means predicting *literal* for the expressions which consist of more literal examples and *nonliteral* for the expressions consisting of more nonliteral examples. We notice the baseline gets a fairly high performance (Acc.=91.86%).

The results show that the expressions can be classified relatively reliably by the proposed features. The performance beats the majority baseline statistically significantly ($p = 0.01$, χ^2 test). The most effective feature combination is the semantic relatedness features and the global context (glc+dc). We notice that *parSyn*, *chiSyn*, *locCont*, *iTerm* and *quote* features are too sparse. These individual features cannot guide the classifier. Therefore, the classifier only predicts the majority class which results in a performance similar to the baseline. Some of the syntactic features are less sparse and they achieve different results from the baseline classifier, however, the performances of these features are actually worse than the baseline. This may be due to the relatively small training size in each idiom specific model. When adding those features together with statistical-based features (glc+dc), the performance of the literal class can be improved slightly. However, we donnot observe any performance increase on the accuracy.

feature	Avg. literal Prec.	Rec.	F-Score	Avg. Acc.
all	89.59	65.77	73.22	89.90
glc+dc	82.53	60.86	70.06	89.08
allSyn	50.83	59.88	54.99	79.42
heaSyn	50.57	59.88	54.83	79.29
sibSyn	33.33	0.86	1.67	78.83
ne	62.45	20.00	30.30	80.69
iTerm	40.00	0.25	0.49	78.99
Base_{maj}	–	–	–	79.01

Table 3.2: Performance of the generic model (averaged over different idioms), 10-fold stratified cross-validation.

3.4.2 Generic Models

Having verified that literal and idiomatic usages can be distinguished with some success by training expression-specific models, we carry out a second experiment in which we merge the data sets for different expressions and train one generic model. We want to see whether a generic model, which has access to more training data, performs better and whether some features, e.g., the cohesion features profit more from this. The experiment was again run in 10-fold stratified cross-validation mode (using 10% from each idiom in the test set in each fold).

Table 3.2 shows the results. The baseline classifier always predict the majority class 'nonliteral'. Note that the result of this baseline is different from the majority baseline in the idiom specific model. In the idiom specific model, there are three expressions (i.e., *bounce off the wall*, *drop the ball*, *pull the trigger*) for which the majority class is 'literal'.

Unsurprisingly, the F-Score and accuracy of the combined feature set drops a bit. However, the performance still statistically significantly beats the majority baseline classifier ($p << 0.01$, χ^2 test). Similar to previous observation, the statistical-based features (glc+dc) work the best, while the syntactic features are also helpful. However, the parSyn, chiSyn, locCont, and quote features are very sparse and, as in the idiom-specific experiments, the performances of these features are similar to the majority baseline classifier. We exclude them from the Table 3.2.

The numbers show that the syntactic features help more in this model compared with the idiom-specific model. When including these features, literal F-Score increases by 3.16% while accuracy increases by 0.9%. It seems that the syntactic features benefit from the increased

	Avg. literal			Avg.
feature	Prec.	Rec.	F-Score	Acc.
all	96.70	81.65	88.54	95.41
glc+dc	96.93	77.00	85.83	94.48
allSyn	52.54	58.77	55.48	79.52
heaSyn	51.35	59.47	55.11	78.96
sibSyn	55.56	2.32	4.46	78.38
ne	61.89	19.05	29.13	79.87
iTerm	66.67	0.7	1.38	78.36
Base_{maj}	–	–	–	79.01

Table 3.3: Performance of the generic model on unseen idioms (cross validation, instances from each idiom are chosen as test set for each fold)

training set. This is evidence that these features can generalize across idioms. For instance, the phrase "The US" on the subject position may be not only indicative of the idiomatic usage of *break the ice*, but also of idiomatic usage of *drop the ball*.

We find that the indicative terms are rare in our corpus. This is the reason why the recall rate of the indicative terms is very low (0.25%). The indicative terms are not very predictive of literal or nonliteral usage, since the precision rate is also relatively low (40%), which means those words can be used in both literal and nonliteral cases.

3.4.3 Unseen Idioms

In our final experiment, we test whether a generic model can also be applied to completely new expressions, i.e., expressions for which no instances have been seen in the training set. Such a behaviour would be desirable for practical purposes as it is unrealistic to label training data for each idiom the model might possibly encounter in a text. To test whether the generic model does indeed generalize to unseen expressions, we test it on all instances of a given expression while training on the rest of the expressions in the dataset. That is, we use a modified cross-validation setting, in which each fold contains instances from one expression in the test set. Since our dataset contains 13 expressions, we run a 13-fold cross validation. The final confusion matrix is the sum over each confusion matrix in each round.

The results are shown in Table 3.3. Similar to the generic model, we find that the cohesion features and syntactic features do generalize across expressions. Statistical features (glc+dc)

	literal F-S.		Acc.	
feature	Spe.	Gen.	Spe.	Gen.
all	86.85	**91.79**	80.67	**88.37**
glc+dc	86.75	**88.84**	80.67	**84.61**
allSyn	**85.71**	71.94	**75.28**	61.13
heaSyn	**85.79**	71.94	**75.39**	61.13

Table 3.4: Comparing the performance of the idiom *drop the ball* on the idiom specific model (Spe.) and generic model (Gen.)

perform well in this experiment. When including more linguistically orientated features, the performance can be further increased by nearly 1%. In line with former observations, the sparse features mentioned in the former two experiments (parSyn, chiSyn, locCont and quote) also do not work for this experiments. We also exclude them from the table.

One interesting finding of this model is that the F-Score is higher than for the "generic model". This is counter-intuitive, since in the generic model, each idiom in the testing set has examples in the training set, thus, we might expect the performance to be better due to the fact that instances from the same expression appearing in the training set are more informative compared with instances from different idioms. Further analysis reveal that there are some expressions for which it may actually be beneficial to train on other expressions by the features of common properties (e.g., semantic relatedness features).

Table 3.4 shows the comparison of the performance of *drop the ball* on the idiom specific model and the generic model on unseen idioms. It can be seen that the statistical features (glc+dc) work better for the model that is trained on the instances from other idioms than the model which is trained on the instances of the target expression itself. We find this is due to the fact that *drop the ball* is especially difficult to classify with the discourse cohesion features (dc). The literal cases are often found in a context containing words, such as **fault**, **mistake**, **fail**, and **miss**, which are often used to describe a scenario in a baseball game,[1] while, on the other hand, those context words are also closely semantically related to the idiomatic reading of *drop the ball*. This means the classifier can be mislead by the cohesion features of the literal instances of this idiom in the training set, as they exhibit strong idiomatic cohesive links with the target expression. When excluding *drop the ball* from the training set, the cohesive links in the

[1]The corpus contains many sports news text.

training data are less noisy. Thus, the performance increases. Unsurprisingly, the performance of syntactic features works better for the idiom specific model compared with the unseen idiom model.

3.5 Related Work

Until recently, most studies on idiom classification focus on type-based extraction (detect idioms on the type level). Type-based methods frequently exploit the fact that idioms have a number of properties which differentiate them from other expressions. For example, they often exhibit a degreee of syntactic and lexical fixedness. Some idioms, for instance, do not allow internal modifiers (**kick the black bucket*) or passivisation (**the bucket was kicked*). They also typically only allow very limited lexical variation (**kick the barrel*, **hit the bucket*). Many approaches for identifying idioms focus on one of these two aspects. For instance, measures that compute the association strength between the elements of an expression have been employed to determine its degree of compositionality (Fazly and Stevenson, 2006; Lin, 1999) (see also Villavicencio et al. (2007) for an overview and a comparison of different measures). Other approaches use Latent Semantic Analysis (LSA) to determine the similarity between a potential idiom and its components (Baldwin et al., 2003). Low similarity is supposed to indicate low compositionality. Bannard (2007) looks at the syntactic fixedness of idiomatic expressions, i.e., how likely they are to take modifiers or be passivised, and compares this to what would be expected based on the observed behaviour of the component words. Fazly and Stevenson (2006) combine information about syntactic and lexical fixedness (i.e., estimated degree of compositionality) into one measure.

Sofar there are only comparably few studies on token-based classification (given a potentially idiomatic phrase in context, decide whether it is used literally or idiomatically). Hashimoto et al. (2006) present a rule-based system in which lexico-syntactic features of different idioms are hard-coded in a lexicon and then used to distinguish literal and nonliteral usages. The features encode information about the passivisation, argument movement, and the ability of the target expression to be negated or modified. Later on, they extend their features (e.g., *collocations*) inspired by other word sense disambiguation tasks and gain performance (Hashimoto and Kawahara, 2008). Katz and Giesbrecht (2006) compute meaning vectors for literal and nonliteral examples in the training set and then classify test instances based on the closeness of their meaning vectors to those of the training examples. This approach is later extended by Diab and Krishna

(2009), which takes a larger context into account (e.g., the whole paragraph), and includes prepositions and determiners in addition to the previous content words. Cook et al. (2007) and Fazly et al. (2009) take a different approach, which crucially relies on the concept of *canonical form* (CForm). It is assumed that for each idiom there is a fixed form (or a small set of those) corresponding to the syntactic pattern(s) in which the idiom normally occurs (Riehemann, 2001).[1] The canonical form allows for inflectional variation of the head verb but not for other variations (such as nominal inflection, choice of determiner etc.). In their work, canonical forms are determined automatically using a statistical, frequency-based measure. Birke and Sarkar (2006) model literal vs. nonliteral classification as a word sense disambiguation task and use a clustering algorithm which compares test instances to two seed sets (one with literal and one with nonliteral expressions), and assign the label of the closest set. Li (2008) and (Sporleder and Li, 2009) propose another unsupervised method which detects the presence or absence of cohesive links between the component words of the idiom and the surrounding discourse. If such links can be found the expression is classified as literal otherwise as nonliteral. Boukobza and Rappoport (2009) experiment with a supervised classifier which takes into account various surface features such as word co-occurrence.

3.6 Summary

In this chapter, we focus on developing a supervised approach to model a two sense category lexical ambiguity problem, i.e., token-based idiom detection. For this task, the classes are fairly imbalanced, with one class (typically the nonliteral interpretation) being much more frequent than the other. This causes problems for training data generation. For idiom specific classifiers, it is difficult to obtain large data sets even when extracting from large corpora and it is even more difficult to find sufficient examples of the minority class. In order to address this problem, we look for features which can generalize across idioms.

We find that statistical features based on semantic relatedness and global context work best for distinguishing literal and nonliteral readings. More specifically, the most effective four individual features that we discovered are *Salient Words*, *Related Words*, *Relatedness Score* and *Discourse Connectivity*. Those most effective features are further used by our successive work in the next two chapters. We find that certain linguistically motivated features can further boost the performance. However, those linguistic features are more likely to suffer from data

[1] This is also the form in which an idiom is usually listed in a dictionary.

sparseness, as a result, they often only predict the majority class if used on their own. We also find that some of the features that we designed generalize well across idioms. The cohesion features have the best generalization ability, while syntactic features can generalize to some extent.

4

A Bootstrapping Model to Disambiguate Idiomatic Expressions

In the last chapter, we adopt a supervised model for the token-based idiom detection task. We experiment with statistically and linguistically informed features and determine the most effective feature combination (salW, relW, relS and conncetivity). The proposed supervised model outperforms a state-of-the-art model, however, the drawback of this model is that it is supervised and large amount of human annotation work is inevitable. In this chapter, we aim to reduce the human annotation effort by introducing an unsupervised model while maintaining a comparable high performance. More specifically, the unsupervised classifier proposed in this chapter is a bootstrapping model which relies on two component classifiers from previous work: the cohesion graph classifier and the supervised model.

4.1 Introduction

Li (2008); Sporleder and Li (2009) describe a cohesion graph method that exploits the presence or absence of cohesive ties between the component words of a potential idiom and its context to distinguish between literal and non-literal use. If strong ties can be found the expression is classified as literal otherwise as non-literal. While this approach often works fairly well, it has the disadvantage that it focuses exclusively on lexical cohesion, other statistical and linguistic cues that might influence the classification decision are disregarded. In their experimental result, they report a (literal) F-Score of 58.26%.

In contrast, the supervised classifier described in Chapter 3 explores more statistical and

linguistic features compared with the cohesion graph method. With further assistance by the human annotation data, it achieves a (literal) F-Score of 73.22%, statistically significantly outperforming the cohesion graph approach ($p << 0.01$, χ^2 test). However, the disadvantage is that it is supervised which needs extra human annotation work compared to the cohesion graph based approach.

In this chapter we show that it is plausible to combine the two classifiers into one framework which takes the advantages from both sides: unsupervised and high performance. We propose a bootstrapping approach, in which the cohesion graph classifier and the supervised classifier serve as component classifiers. We use the unsupervised classifier to label a sub-set of the test data with high confidence. This sub-set is then passed on as training data to the supervised classifier, which then labels the remainder of the data set. This process goes on iteratively as more and more instances are selected to the training set and predicted by the bootstrapping framework. Compared to the cohesion graph approach, this approach has the advantage that a larger feature set can be exploited. This is beneficial for examples, in which the cohesive ties are relatively weak but which contain other linguistic cues for literal or non-literal use. Compared to the supervised model, this approach has the advantage that it does not need any human annotated data.

4.2 Component Classifiers

In this section, we introduce two component classifiers on which the bootstrapping model is built up: one unsupervised classifier and one supervised classifier.

4.2.1 Unsupervised Classifier

As our unsupervised classifier, we use the cohesion graph (Li, 2008; Sporleder and Li, 2009) (see Section 3.3.1 for a short description). We hypothesize that the unsupervised classifier gives us relatively good results for some examples. For instance, in (3.4) there are several strong cues which suggest that *play with fire* is used literally. However, because the unsupervised classifier only looks at lexical cohesion, it misses many other clues which could help distinguish literal and non-literal usages. For example, if *break the ice* is followed by the prepositions *between* or *over* as in example (4.1), it is more likely to be used idiomatically (at least in the news domain).

(4.1) "Gujral will meet Sharif on Monday and discuss bilateral relations," the Press Trust of India added. The minister said Sharif and Gujral would be able to break the ice over Kashmir.

Furthermore, idiomatic usages also exhibit cohesion with their context but the cohesive ties are with the *non-literal* meaning of the expression. For example, in news texts, *break the ice* in its figurative meaning often co-occurs with *discuss*, *relations*, *talks* or *diplomacy* (see (4.1)). Due to the reasons described in Section 3.2 we do not have effective way to model these idiomatic cohesive links. However if we label data we can train a supervised classifier to learn these and other contextual clues. The trained classifier may then be able to correctly classify examples which were misclassified by the unsupervised classifier, i.e., examples in which the cohesive ties are weak but where other clues exist which indicate how the expression is used.

For example, in (4.2) there is weak cohesive evidence for a literal use of *break the ice*, due to the semantic relatedness between *ice* and *water*. However, there are stronger cues for non-literal usage, such as the preposition *between* and the presence of words like *diplomats* and *talks*, which are indicative of idiomatic usage. Examples like this are likely to be misclassified by the unsupervised model; a supervised classifier, on the other hand, has a better chance to pick up on such additional cues and predict the correct label.

(4.2) Next week the two diplomats will meet in an attempt to break the ice between the two nations. A crucial issue in the talks will be the long-running water dispute.

4.2.2 Supervised Classifier

For the supervised classifier, we use the one described in Chapter 3. We select the most effective four features (salW, relW, relS, and connect.)[1], which encode both lexical cohesion and word co-occurrence information.[2] We use SVM implemented as the LIBSVM package.[3]

[1] Refer to Chapter 3 for descriptions of these four features.

[2] As described in the last chapter, we also experiment with linguistically more informed features, such as the presence of named entities in the local context of the expression, and properties of the subject or co-ordinated verbs, but we find that these features do not lead to a better performance of the supervised classifier. Thus, we decide to only include the most effective four features for computation efficiency.

[3] Available from: `http://www.csie.ntu.edu.tw/~cjlin/libsvm/` We used the default parameters. We choose this package, because the Java API can be easily integrated in our boostrapping code.

4.3 Bootstrapping

The bootstrapping process starts with the unsupervised classifier labeling a small initial training set, which, consequently, is used to train the supervised classifier. The supervised classifier applies the trained model to the rest of the data and predicts the labels. Then we select the most confident examples from the prediction to argument the training size of the supervised classifier in the next round. The whole process goes iteratively as more and more confident examples are included in the training set, and it stops after predefined number of iterations is reached. To ensure that the training set does not contain too much noise, we only add those examples about which the unsupervised classifier is most confident. We thus need to address two questions: (i) how to define a *confidence function* for the bootstrapping, and (ii) how to set the *confidence threshold* governing what proportion of the data set is used for training the supervised classifier in the next round.

As the unsupervised classifier bases its decision on the difference in connectivity between including or excluding the component words of the idiom in the cohesion graph, one possible choice for a confidence function is the difference in connectivity; i.e., the higher the difference, the higher the confidence of the classifier in the predicted label. The confidence threshold could be selected on the basis of the unsupervised classifier's performance on a development set. Note that when choosing such a threshold there is usually a trade-off between the size of the training set and the amount of noise in it: the lower the threshold, the larger and the noisier the training set. Ideally we would like a reasonably-sized training set which is also relatively noise-free, i.e., does not contain too many wrongly labeled examples. One way to achieve this is to start with a relatively small training set and then expand it gradually.

A potential problem for the supervised classifier is that our data set is relatively imbalanced, with the non-literal class being four times as frequent as the literal class. Supervised classifiers often have problems with imbalanced data and tend to be overly biased towards the majority class (see, e.g., Japkowicz and Stephen (2002)). To overcome this problem, we experiment with boosting the literal class with additional examples.[1]

Figure 4.1 describes the whole bootstrapping process by showing different components and how they connect with each other. The main two components of the bootstrapping consist of two parts: i) the iterative training module, and ii) the boosting literal class module. We discuss them in the next two sections.

[1]Throughout this paper, we use the term 'boosting' in a non-technical sense.

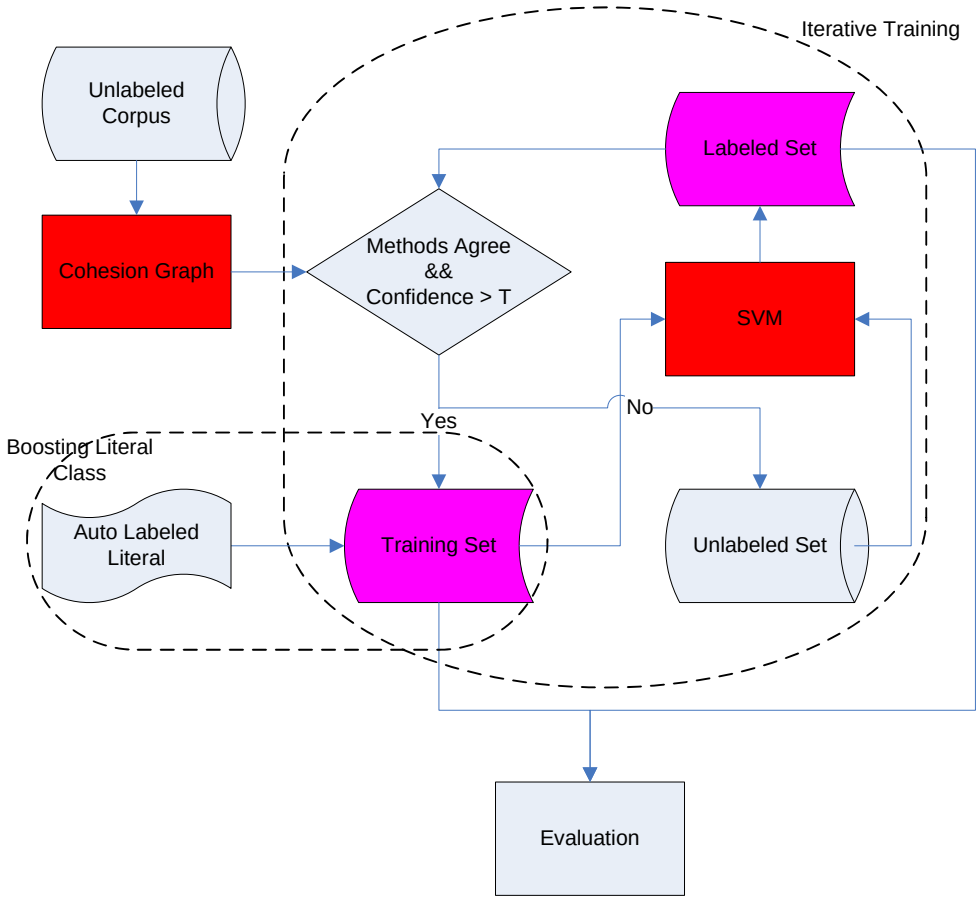


Figure 4.1: The bootstrapping classifier.

4.3.1 Iteratively Enlarging the Training Set

A typical method for increasing the training set is to go through several iterations of enlargement and re-training.[1] We adopt a conservative enlargement strategy: we only consider instances on whose labels both classifiers agree and we use the confidence function of the unsupervised classifier to determine which of these examples to add to the training set. The motivation for this is that we hypothesize that the supervised classifier does not have a very good performance initially, as it is trained on a very small data set. As a consequence its confidence function may also not be very accurate. On the other hand, we know from previous work that the unsupervised

[1] In our case re-training also involves re-computing the ranked lists of salient and related words. As the process goes on the classifier will be able to discover more and more useful cue words and encode them in the feature vector.

classifier has a reasonably good performance. So while we give the supervised classifier a veto-right, we do not allow it to select new training data by itself or overturn classifications made by the unsupervised classifier.

A similar strategy was employed by Ng and Cardie (2003) in a self-training set-up. However, while they use an ensemble of supervised classifiers, which they re-train after each iteration, we can only re-train the second classifier; the first one, being unsupervised, will never change its prediction. Hence it does not make sense to go through a large number of iterations; the more iterations we go through, the closer the performance of the combined classifier will be to that of the unsupervised one because that classifier will label a larger and larger proportion of the data. However, going through a few iterations allows us to slowly enlarge the training set and thereby gradually improve the performance of the supervised classifier.

In each iteration, we select 10% of the remaining examples to be added to the training set.[1] We could simply add those 10% of the data about which the unsupervised classifier is most confident, but if the classifier was more confident about one class than about the other, we would risk obtaining a severely imbalanced training set. Hence, we decided to separate examples classified as 'literal' from those classified as 'non-literal' and add the top 10% from each set. Provided the automatic classification is reasonably accurate, this ensures that the distribution of classes in the training set is roughly similar to that in the overall data set at least at the early stages of the bootstrapping.

4.3.2 Boosting the Literal Class

As the process goes on, we are still likely to introduce more and more imbalance in the training set. This is due to the fact that the supervised classifier is likely to have some bias towards the majority class (and our experiments in Section 4.4.1 suggest that this is indeed the case). Hence, as the bootstrapping process goes on, potentially more and more examples will be labeled as 'non-literal' and if we always select the top 10% of these, our training set will gradually become more imbalanced. This is a well-known problem for bootstrapping approaches (Le et al., 2006; **?**). We could counteract this by selecting a higher proportion of examples labeled as 'literal'. However given that the number of literal examples in our data set is relatively small, we would soon deplete our literal instance pool and moreover, because we would be forced to add less

[1] 10% is the number used for evaluating the final bootstrapping classifier. Before that, we also experiment with different confidence thresholds. The results are reported in Section 4.4.2.

confidently labeled examples for the literal class, we are likely to introduce more noise in the training set.

A better option is to boost the literal class with external examples. To do this we exploit the fact that non-canonical forms of idioms are highly likely to be used literally. Given that our data set only contains canonical forms (see Sporleder and Li (2009)), we automatically extract non-canonical form variants and label them as 'literal'. To generate possible variants, we either (i) change the number of the noun (e.g., *rock the boat* becomes *rock the boats*), (ii) change the determiner (e.g., *rock a boat*), or (iii) replace the verb or noun by one of its synonyms, hypernyms, or siblings from WordNet (e.g., *rock the ship*). While this strategy does not give us additional literal examples for all idioms, for example we were not able to find non-canonical form occurrences of *sweep under the carpet* in the Gigaword corpus, for most idioms we were able to generate additional examples. Note that this data set is potentially noisy as not all non-canonical form examples are used literally. However, when checking a small sample manually, we find that only very small percentage ($<< 1\%$) was mis-labelled.

To reduce the classifier bias when enlarging the training set, we add additional literal examples during each iteration to ensure that the class distribution does not deviate too much from the distribution originally predicted by the unsupervised classifier.[1] The examples to be added are selected randomly but we try to ensure that each idiom is represented. When reporting the results, we disregard these additional external examples.

4.4 Experiments

We carry out three different experiments: (i) In Section 4.4.1 we investigate the performance of the individual features of the supervised classifier; (ii) In Section 4.4.2 we test how the confidence function influence the performance of the classifiers; and (iii) In the last Section 4.4.3 we look more closely at the behaviour of the bootstrapping classifier.

4.4.1 Feature Analysis for the Supervised Classifier

In the first experiment, we investigate the performance of the different features of the supervised classifier (Table 4.1). For each set, we trained a separate classifier and tested it in 10-fold cross-validation mode. We also tested the performance of the first three features combined (salient

[1] We are assuming that the true distribution is not known and use the predictions of the unsupervised classifier to approximate the true distribution.

	Avg. literal (%)			Avg. (%)
Feature	Prec.	Rec.	F-Score	Acc.
salW	77.10	56.10	65.00	86.83
relW	78.00	43.20	55.60	84.99
relS	74.90	37.50	50.00	83.68
connectivity	78.30	2.10	4.10	78.58
salW+relW+relS	82.90	63.50	71.90	89.20
all	85.80	66.60	75.00	90.34

Table 4.1: Performance of different feature sets, 10-fold cross-validation

and related words and relatedness score) as we wanted to know whether their combination leads to performance gains over the individual classifiers. Moreover, testing these three features in combination allows us to assess the contribution of the connectivity feature, which is most closely related to the unsupervised classifier. We report the accuracy, and because our data are fairly imbalanced, also the F-Score for the minority class ('literal').

It can be seen that the *salient words* (*salW*) feature has the highest performance of the individual features, both in terms of accuracy and in terms of literal F-Score, followed by *related words* (*relW*), and *relatedness score* (*relS*). Intuitively, it is plausible that the saliency feature performs quite well as it can also pick up on linguistic indicators of idiom usage that do not have anything to do with lexical cohesion. However, a combination of the first three features leads to an even better performance, suggesting that the features do indeed model different aspects of the data. The performance of the connectivity feature is also interesting: while it does not perform very well on its own, as it over-predicts the non-literal class, it noticeably increases the performance of the model when combined with the other features, suggesting that it picks up on complementary information.

4.4.2 Effects of the Confidence Threshold

The performance of the methods depends on finding a good confidence threshold for the bootstrapping. To determine how sensitive our method is to this parameter we ran experiments in which we vary the confidence threshold in 200 steps from 0.005 to 1 and record the results.

Figure 4.2 represents the performance of the unsupervised examples on the top percent confident predictions. As shown in the figure, the performance goes down as less confident examples are included. The exception cases are the first 5-6%. However, those exceptions

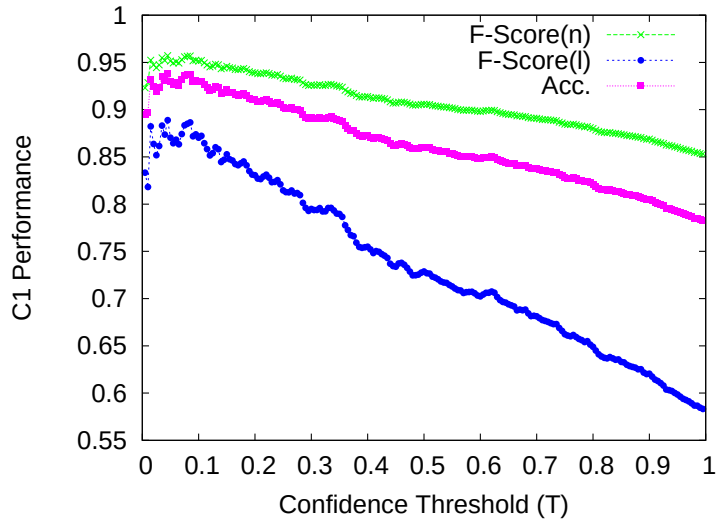


Figure 4.2: Performance of the unsupervised classifier on top percent confident examples. F-Score(n) is the F-Score of the nonliteral class, F-Score(l) is the F-Score of the literal class, Acc. is accuracy, C1 is the unsupervised classifier.

occur on a very small sample size (top 5% contains about 200 examples). The curve is more informative at the part where the instance size is relatively big (percentage $>$ 10%). Table 4.2 shows the details of the output of the top confident examples by the cohesion graph classifier. The accuracy (Acc.) of the top 5 confident predictions is 92.93%, while it is 89.13% for the top 30% confident examples. This experiment shows that the selected confidence function is a good indicator of the true label, which justifies our decision to choose the discourse connectivity difference as the confidence function for the bootstrapping model.

Figure 4.3a shows the performance of the supervised classifier if trained on the output of the unsupervised classifier. It can be seen that this classifier's performance initially goes up as there are more number of instance in the training set. Then, the performance drops. If 90% of the data are labeled by the unsupervised classifier, the performance of the supervised calssifier on the remaining 10% drops to less than 40% (literal F-Score). With a higher confidence threshold, the supervised classifier benefits from more training data but these benefits seem to be more than outweighed by the increased noise in the examples labels so that the overall performance drops. Additionally it can also be that the examples not confidently predicted by the unsupervised classifier are simply the most difficult ones, and with an increasing confidence threshold this leaves fewer and fewer of the easier examples for the supervised classifier.

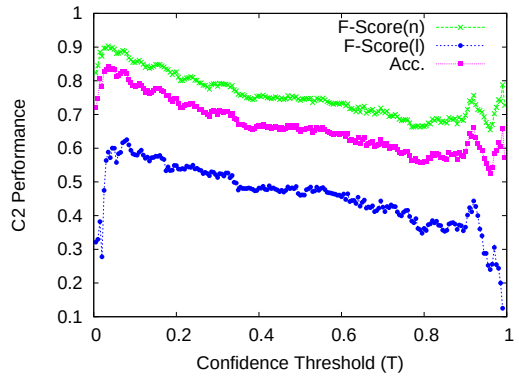
TopConf.(%)		output$_n$	output$_l$	Pre.	Rec.	F$_{\beta=1}$	Acc.
5	label$_n$	137	13	99.28	91.33	95.14	92.93
	label$_l$	1	47	78.33	97.92	87.04	
20	label$_n$	547	66	99.09	89.23	93.91	91.04
	label$_l$	5	174	72.50	97.21	83.05	
30	label$_n$	809	110	97.71	88.03	92.62	89.14
	label$_l$	19	250	69.44	92.94	79.49	

Table 4.2: Performance of the top confident examples. TopConf. is the top percentage predictions; output$_n$ is the number of examples predicted as "nonliteral"; output$_l$ is the number of examples predicted as "literal"; label$_n$ represents labelled as "nonliteral" in the gold standard; Pre. is precision; Rec. is recall; F is F-Score; Acc. is accuracy.

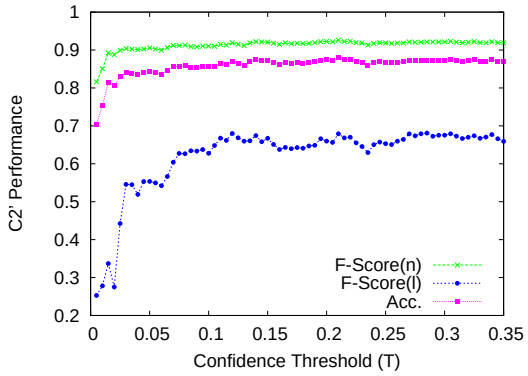
For comparison, Figure 4.3b and 4.3c show the performance of the supervised classifier that would be obtained with gold standard labels from the unsupervised one. This curve goes up initially (see Figure 4.3b), then stays at around the same level for while, and in the end drops (Figure 4.3c). This suggest that the second stage classifier actually reaches its peak after seeing about 15% of the training data. Increasing the training set further does not seem to lead to improvements. For the performance of the last 10%, the performance actually drops even if it uses more than 90% examples as training data. Remember that those last 10% are the examples that are the least confident by the unsupervised examples. This further suggest that the unsupervised and supervised classifier agree on the difficult cases.[1]

Finally, Figure 4.4 shows the overall performance of directly combining the two classifiers (also can be seen as bootstrapping with only one iteration). As seen from the Figure, the performance of the combined classifier reaches to its peek when around 8% of the examples are labeled by the unsupervised classifier and the rest 92% are labeled by the supervised classifier (trained on the 8% labeled previous). Further increase the size of the training set of the supervised classifier does not lead to improvement, as more and more noise data are introduced in the training data. This study suggest that our bootstrapping model should find a right strategy to balance the training size and label errors in the training set. In practice, we experimented with different combinations, the final results reported of the bootstrapping model are based on the optimal combination.

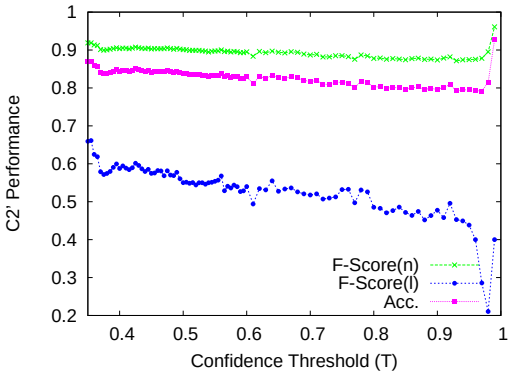
[1]This is actually not supervising, as the two classifiers share the similar feature, semantic relatedness.



(a) Without label correction.



(b) With label correction.



(c) With label correction

Figure 4.3: Performance of the supervised classifier trained on top % confident examples output by the unsupervised classifier (with/without label correction), bootstrapping one iteration. F-Score(n) is the F-Score of the nonliteral class, F-Score(l) is the F-Score of the literal class, Acc. is Accuracy, c2 (supervised classifier).

4.4.3 Testing the Bootstrapping Classifier

We experiment with different variants of the bootstrapping classifier. The results are shown in Table 4.3. In particular, we look at: (i) combining the two classifiers without training set enlargement or boosting of the literal class (*combined*), (ii) boosting the literal class with 200 automatically labelled non-canonical form examples (*combined+boost*), (iii) enlarging the training set by iteration (*combined+it*), and (iv) enlarging the training set by iteration and boosting the literal class after each iteration (*combined+boost+it*). The table shows the literal precision, recall and F-Score of the combined model (both classifiers) on the complete data set

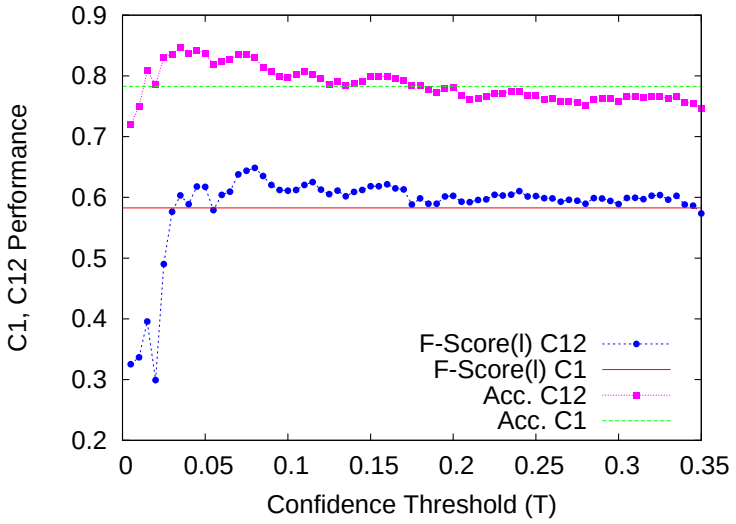


Figure 4.4: Performance of directly combining the two classifiers based on different confidence threshold. F-Score(n) is the F-Score of the nonliteral class, F-Score(l) is the F-Score of the literal class, Acc. is Accuracy, c1 (unsupervised classifier), c12 (bootstrapping classifier).

(excluding the extra literal examples). Note that the results for the set-ups involving iterative training set enlargement are optimistic: since we do not have a separate development set, we report the optimal performance achieved during the first seven iterations. In a real set-up, when the optimal number of iterations is chosen on the basis of a separate data set, the results may be lower. The table also shows the majority class baseline ($Base_{maj}$), and the overall performance of the unsupervised model (*unsup*) and the supervised model when trained in 10-fold cross-validation mode (*super 10CV*).

It can be seen that the combined classifier is 8% more accurate than both the majority baseline and the unsupervised classifier. This amounts to an error reduction of over 35% (the difference is statistically significant, χ^2 test, $p << 0.01$). While the F-Score of the unboosted combined classifier is comparable to that of the unsupervised one, boosting the literal class leads to a 7% increase, due to a significantly increased recall, with no significant drop in accuracy. These results show that complementing the unsupervised classifier with a supervised one, can lead to tangible performance gains. Note that the accuracy of the combined classifier, which uses no manually labelled training data, is only 4% below that of a fully supervised classifier; in other words, we do not lose much by starting with an automatically labelled data set. Iterative enlargement of the training set can lead to further improvements, especially when combined

Model	Prec_l	Rec_l	F-Score_l	Acc.
Base_{maj}	-	-	-	78.25
unsup.	50.04	69.72	58.26	78.38
combined	83.86	45.82	59.26	86.30
combined+boost	70.26	62.76	66.30	86.13
combined+it*	85.68	46.52	60.30	86.68
combined+boost+it*	71.86	66.36	69.00	87.03
super. 10CV	85.80	66.60	75.00	90.34

Table 4.3: Results for different classifiers; $*$ indicates best performance (optimistic)

with boosting to reduce the classifier bias.

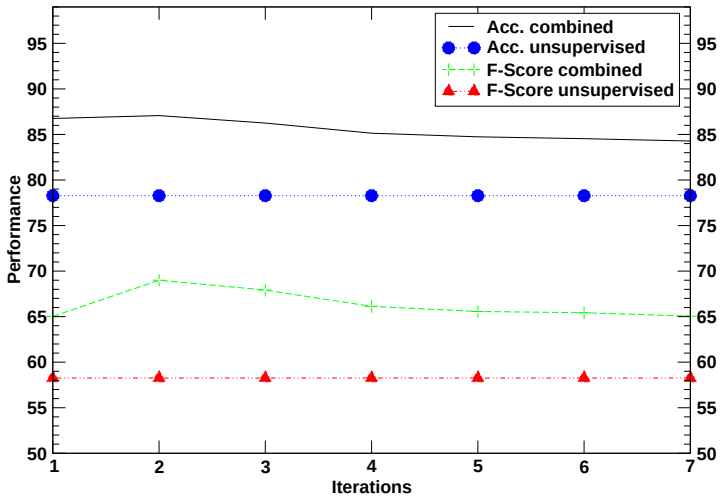


Figure 4.5: Accuracy and literal F-Score on complete data set after different iterations with boosting of the literal class, 'combined' is the bootstrapping model.

To get a better idea of the effect of training set enlargement, we plot the accuracy and F-Score of the bootstrapping classifier for a given number of iterations with boosting (Figure 4.5) and without (Figure 4.6). It can be seen that enlargement has a noticeable positive effect if combined with boosting. If the literal class is not boosted, the increasing bias of the classifier seems to outweigh most of the positive effects from the enlarged training set. Figure 4.5

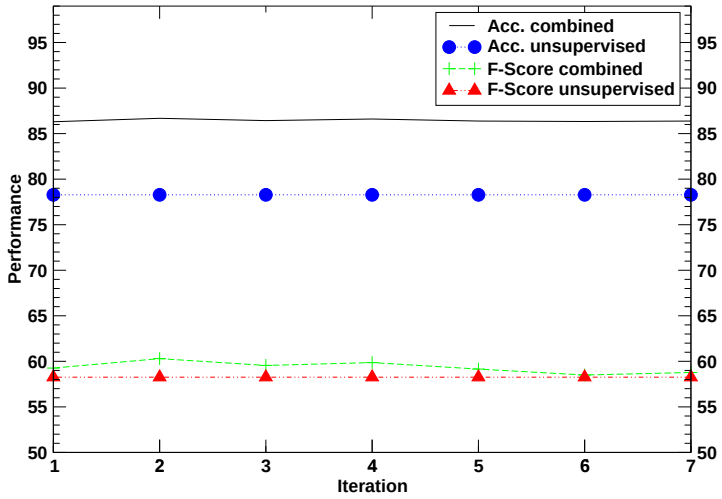


Figure 4.6: Accuracy and literal F-Score on complete data set after different iterations without boosting of the literal class, 'combined' is the bootstrapping model.

also shows that the best performance is obtained after a relatively small number of iterations (namely two), as expected.[1] With more iterations the performance decreases again. However, it decays relatively gracefully and even after seven iterations, when more than 40% of the data are included in the training set, the bootstrapping classifier still achieves an overall performance that is significantly above that of the unsupervised classifier (84.28% accuracy compared to 78.38%, significant at $p << 0.01$). Hence, the bootstrapping classifier seems not to be very sensitive to the exact number of iterations and performs reasonably well even if the number of iterations is sub-optimal.

Figure 4.7 shows how the training set increases as the process goes on[2] and how the number of mis-classifications in the training set develops. Interestingly, when going from the first to the second iteration the training set nearly doubles (from 396 to 669 instances), while the proportion of errors is also reduced by a third (from 7% to 5%). Hence, the training set does not only grow but the proportion of noise in it decreases, too. This shows that our conservative enlargement strategy is fairly successful in selecting correctly labeled examples. Only at later stages, when

[1] Note that this also depends on the confidence threshold. For example, if a threshold of 5% is chosen, more iterations may be required for optimal performance.

[2] Again, we disregard the extra literal examples here.

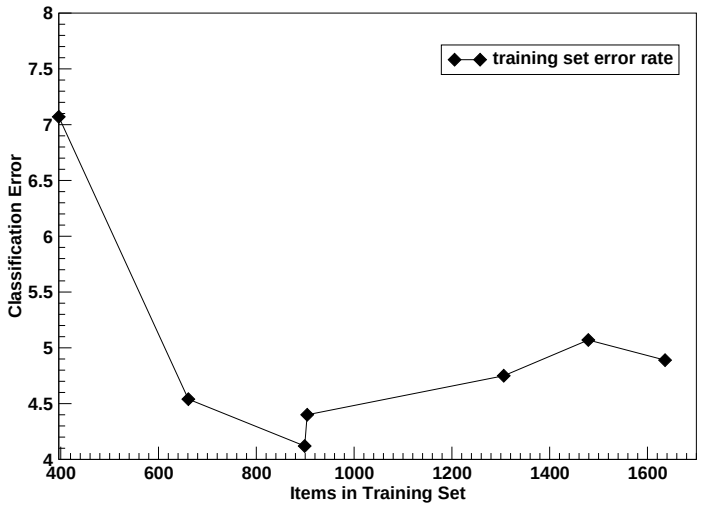


Figure 4.7: Training set size and error in training set at different iterations

the classifier bias takes over, does the proportion of noise increase again.

4.5 Related Work

Bootstrapping is process in which a handful of labeled instances are used to initially train a classifier. Then, the classifier is applied to the rest of the data set to predict the labels. Consequently, those labeled examples are selected to be trustworthy or untrustworthy by certain confidence criteria, in which the trustworthy examples are included in the labeled set in the next round. The process goes on iteratively until certain stopping criteria is fulfilled (e.g., enough instances are labeled).

Bootstrapping is adopted in various lexical semantics tasks. Yarowsky (1995) is one of the first few well-known bootstrapping algorithms in this field. The work proposes a bootstrapping approach for Word Sense Disambiguation. This process relies on an initial small size training data which is used for training a supervised classifier. Then, the supervised classifier labels more instances which, according to two selection criteria (*one sense per collocation* and *one sense per discourse*), can be selected to be included in the labeled set in the next round. The process iteratively goes on as more and more instances are labeled. The paper reports the proposed procedure outperforms the previous supervised methods. Later on, Abney (2004) analyze the

Yarowsky paper from a mathematically point. He formulates the optimizing objective function of the bootstrapping algorithm, and proposes a number of variants of the Yarowsky algorithm which are optimize on different objective functions. Subsequent research in this area includes the work by Mihalcea and Moldovan (2001), which relies on WordNet and a sense tagged corpus as the initial set, and tags new instances based on their relation to the already disambiguated set. This work reports an accuracy of 92%. Further in this direction, bootstrapping is shown to be effective in bilingual word sense disambiguation by Li and Li (2004). The approach builds a bootstrapping process which utilizes information from bilingual corpus and achieves a significant improvement over the state-of-the-art monolingual bootstrapping. Bootstrapping is also used for lexicon acquisition task (Riloff and Shepherd, 1997).

In addition to these lexical NLP tasks, bootstrapping is also adopted in various other NLP tasks. Mcclosky et al. (2006) propose a self-training bootstrapping framework for parsing. They report improvement over the previous best result on Wall Street Journal corpus. Blum and Mitchell (1998) propose a co-training framework in which two distinct views on the same dataset can help argument the training set. They successfully utilize this approach to classify web pages. Bellare et al. (2007) propose a lightly-supervised co-training model to extract entity attributes from texts. It is also shown that bootstrapping is effective in Named Entity classification task (Collins and Singer, 1999).

4.6 Summary

In this chapter, we aim to reduce the human annotation effort of the previous chapter by proposing a bootstrapping classification approach for distinguishing literal and non-literal use of idiomatic expressions. Our approach relies on two component classifiers: an unsupervised classifier which exploits information about the cohesive structure of the discourse, and a supervised classifier. The latter can make use of a range of features and therefore base its classification decision on additional properties of the discourse, besides lexical cohesion. We showed that such a hybrid classifier can lead to a significant reduction of classification errors. Its performance can be improved further by iteratively increasing the training set in a bootstrapping loop and by adding additional examples of the literal class, which is typically the minority class. We find that such examples can be obtained automatically by extracting non-canonical variants of the target idioms from an unlabeled corpus.

Future work should look at alternative strategies to score the confidence function in the bootstrapping process. While this is already pretty good, a more sophisticated confidence strategy to select instance included in the training set of the next iteration may lead to further improvement.

5

A Gaussian Mixture Model on Figurative Expression Detection

The idiom task described in the previous two chapters is constrained to idiomatic phrases that are lexicalized and can be found in dictionaries (e.g., *break the ice*, *play with fire*). However, it is also very often the case that certain words or phrases are used nonliterally but they are not lexicalized (e.g., *take the sock out of your mouth* in Example 5.1, and *sparrow* in Example 5.2[1]). In this chapter, we cover another aspects of the study of nonliteral phrases: detect general nonliteral expressions or novel variants of idioms in running texts (token-based unlexicalized figurative expression detection).

(5.1) Take the sock out of your mouth and create a brand-new relationship with your mom.

(5.2) During the Iraq war, he was a sparrow; he didn't condone the bloodshed but wasn't bothered enough to go out and protest.

We focus our study on general figurative expressions which may not be listed in idiom dictionaries but can be used nonliterally in certain contexts. We propose an unsupervised Gaussian Mixture Model to detect those figurative expressions in context. Furthermore, as the idiom task can be seen as a special case of detecting general figurative expression, we also run the GMM on the idiom dataset, and compare the performance of the two to get more insights of the common and distinct aspects of the two related tasks.

[1] We refer the reader to more example usages of this category to Section 2.2.

5.1 Introduction

Figurative language employs words in a way that deviates from their normal meaning. It includes idiomatic usage, metaphor, metonymy or other types of creative language. Being able to detect figurative language is important for a number of NLP applications, e.g., machine translation.

Simple dictionary look-up would not work for truly creative, one-off usages (unlexicalized usages); these can neither be found in a dictionary nor can they be detected by standard idiom extraction methods, which apply statistical measures to accumulated corpus evidence for an expression to assess its 'idiomaticity'. An example of a fairly creative usage is shown as Example 5.1 which is a variation of the idiom *put a sock in*.

In this chapter, we propose a method for detecting figurative language in context. As we use context information rather than corpus statistics, our approach works also for truly creative usages.

5.2 A Gaussian Mixture Model

We model the problem by using Gaussian Mixture Model (GMM). We assume that the literal (l) and non-literal (n) instances are generated by two different Gaussians (*literal* and *nonliteral* Gaussians). The token-based detection task is to compare which Gaussian has the higher probability of generating a specific instance.

The Gaussian mixture model is defined as:

$$p(x) = \sum_{c \in \{l,n\}} w_c \times N(x|\mu_c, \Sigma_c) \tag{5.3}$$

where, c is the category of the Gaussian (*literal* or *nonliteral*), μ_c is the Gaussian mean, Σ_c is the Gaussian covariance matrix, and w_c is the Gaussian weight.

Our method is based on the insight that figurative language exhibits less semantic cohesive ties with the context than literal language (see Chapter 3 and 4 for more details). We use Normalized Google Distance to model semantic relatedness (**?**) and represent the instances by five types of semantic relatedness features $x = (x1, x2, x3, x4, x5)$:

- $x1$ is the average relatedness between the target expression and context words,

$$x1 = \frac{2}{|T| \times |C|} \sum_{(w_i, c_j) \in T \times C} relatedness(w_i, c_j) \tag{5.4}$$

where w_i is a component word of the target expression (T); c_j is one of the context words (C); $|T|$ is the total number of words in the target expression, and $|C|$ is the total number of words in the context. The term $\frac{2}{|T|\times|C|}$ is the normalization factor, which is the total number of relatedness pairs between target component words and context words.

- $x2$ is the average semantic relatedness in the context of the target expression,

$$x2 = \frac{1}{\binom{|C|}{2}} \sum_{(c_i,c_j)\in C\times C, i\neq j} relatedness(c_i, c_j) \tag{5.5}$$

- $x3$ is the difference between the average semantic relatedness between the target expression and the context words and the average semantic relatedness of the context,

$$x3 = x1 - x2 \tag{5.6}$$

It is an indicator of how strongly the target expression is semantically related to the discourse context.

- $x4$ is the same feature used for predicting literal or idiomatic use by the cohesion graph based method (see Chapter 3),

$$x4 = \begin{cases} 1 & \text{if } x3 < 0 \\ 0 & \text{else} \end{cases} \tag{5.7}$$

- $x5$ is a high dimensional vector which represents the top relatedness scores between the component words of the target expression and the context (also called *related score*, see Chapter 4),

$$x5(k) = \max_{(w_i,c_j)\in T\times C}(k, \{relatedness(w_i, c_j)\}) \tag{5.8}$$

where the function $max(k, A)$ is defined to choose the k^{th} highest element from the set A. We set k to be 100 in our experiment.

The detection task is done by a Bayes decision rule, which chooses the category by maximizing the probability of fitting the data into the different Gaussian components (see Equation 5.9). The instance is predicted as figurative if it fits into the nonliteral Gaussian better, and literal if it fits into the literal Gaussian better.

$$c(x) = \arg\max_{i\in\{l,n\}}\{w_i \times N(x|\mu_i, \Sigma_i)\} \tag{5.9}$$

Model	Class	Precision	Recall	F-Score	Accuracy
Co-Graph	n	90.55	80.66	85.32	78.38
	l	50.04	69.72	58.26	
GMM	n	90.69	80.66	85.38	78.39
	l	50.17	70.15	58.50	

Table 5.1: Results on the idiom data set, n(on-literal) is the union of the predefined three sub-classes (nsu, nsa, nw), l(iteral).

5.3 Experiments

In this section, we describe two types of experiments: In Section 5.3.1, we describe the experiments of estimating the Gaussian components by EM algorithm as there is no training data; and in Section 5.3.2, we describe the experiments by estimating the GMM from the labeled data set, the idiom data set (UdSic, see Section 2.1). In both settings, we evaluate our experimental results on the UdSfec (Section 2.2) testing set.

5.3.1 GMM Estimated by EM

We use a MatLab package (Calinon, 2009; Calinon et al., 2007) for estimating the GMM model. The GMM is trained by the EM algorithm. The priors of Gaussian components, means and covariance of each components, are initialized by the k-means clustering algorithm (Hartigan, 1975). We run EM on the combination of the UdSic and UdSfec corpus.

To determine whether the GMM is able to perform token-based idiom classification[1], we applied it to the idiom data set. The results (see Table 5.1) show that the GMM can distinguish usages quite well and gains equally good results as the cohesion graph method (*Co-Graph*) (Li, 2008; Sporleder and Li, 2009). In addition, this method can deal with unobserved occurrences of non-literal language.

Table 5.2 shows the results on the figurative expression data set. We build two baselines. The first baseline (Baseline*) predicts 'idiomatic' and 'literal' based on a uniform random distribution (the chance of predicting 'literal' is the same as the chance of predicting 'idiomatic'.). The second baseline (Baseline) predicts 'idiomatic' and 'literal' according to a biased probability which is based on the distribution in the annotated set. *GMM* shows the performance on the whole data set. We also split the test set into three different subsets to determine how the GMM

[1]Token-based idiom classification is quite similar to figurative expression detection.

Model	Class	Precision	Recall	F-Score	Accuracy
Baseline*	n	17.87	49.33	26.24	50.83
	l	82.41	51.15	63.12	
Baseline	n	21.79	22.67	22.22	71.87
	l	83.19	82.47	82.83	
Co-Graph	n	37.29	84.62	51.76	70.92
	l	95.12	67.83	79.19	
GMM	n	40.71	73.08	52.29	75.41
	l	92.58	75.94	83.44	
GMM{nsu,l}	n	8.79	1.00	16.16	76.49
	l	1.00	75.94	86.33	
GMM{nsa,l}	n	22.43	77.42	34.78	76.06
	l	97.40	75.94	85.34	
GMM{nw,l}	n	23.15	64.10	34.01	74.74
	l	94.93	75.94	84.38	
GMM'	n	22.40	35.90	27.59	65.25
	l	83.22	71.88	77.14	

Table 5.2: Results on the figurative expression data set, Gaussian component parameters estimated by EM

performs on distinguishing literal usage from the different types of figurative usage (Section 2.2): *GMM{nsu, l}*, *GMM{nsa, l}*, *GMM{nw, l}*. We also run an experiment to test how the EM estimation is sensitive to the size of the data. The *GMM'* is the result of running EM purely on the test set (UdSfec).

The unsupervised GMM model beats the baselines and achieves good results on the UdSfec data set. It also outperforms the Co-Graph approach, which suggests that the statistical model,

GMM, is more likely to boost the performance by capturing statistical properties of the data for more difficult cases (*idioms* v.s. *general figurative usages*), compared with the Co-Graph approach.

We also find that GMM is the best at distinguishing unambiguous phrase level figurative usage (nsu, see Section 2.2 for details). The performance on the subset *{nsu, l}* is the highest of all the subset combinations. GMM correctly label all the figurative instances in this experiment (literal precision is 1, and non-literal recall is 1). The reason is that figurative expressions in this category are least likely to exhibit cohesive ties with their surrounding context (e.g. Shall we go *trip the light fantastic*?). In contrast, the most difficult subset is *{nw, l}*. The performance on this subset is the lowest of all. The reason is that the expressions are only partially used figuratively. While some component words of the expression break the cohesion tie with its surrounding context, other component words may maintain this lexical cohesion. As a result, the mixed cohesion features make it more difficult for GMM to decide between *figurative* and *literal* in this experiment. Furthermore, we also find out that more experimental data is good for running EM, as the performance of *GMM'* is very poor (65.25% v.s. 75.41%) when there are only small number of instances available.

In conclusion, the model is not only able to classify idiomatic expressions but also to detect new figurative expressions. However, the performance on the second data set is worse compared with running the same model on the idiom data set. This is because the V+NP data set contains more difficult examples, e.g., expressions which are only partially figurative (e.g., Example 2.8). One would expect the literal part of the expression to exhibit cohesive ties with the context, hence the cohesion based features may fail to detect this type of figurative usage. Consequently the performance of the GMM is lower for figuratively used words ('nw') than for idioms ('nsa', 'nsu'). However, even for figurative words cases ('nw') the model still obtains a relatively high accuracy.

5.3.2 GMM Estimated from Annotated Data

In a second experiment, we test how well the GMM performs when utilizing the annotated idiom data set to estimate the two Gaussian components instead of using EM. We give equal weights to the two Gaussian components and predict the label on the UdSfec data set by fixing the mixture model which is estimated from the UdSic training set (GMM+f). This method further improves the performance compared to the unsupervised approach (Table 5.3).

Model	Class	Precision	Recall	F-Score	Accuracy
GMM+f	n	42.22	73.08	53.52	76.60
	l	92.71	77.39	84.36	
GMM+f+s	n	41.38	54.55	47.06	83.44
	l	92.54	87.94	90.18	

Table 5.3: Results on the figurative expression data set, Gaussian component parameters estimated by annotated data

We also experiment with setting a threshold and abstaining from making a prediction when the probability of an instance belonging to the Gaussian is below the threshold (GMM+f+s). Table 5.3 shows the performance when only evaluating on the subset for which a classification was made. It can be seen that the accuracy and the overall performance on the literal class improve, but the precision for the non-literal class remains relatively low, i.e., many literal instances are still misclassified as 'non-literal'. One reason for this may be that there are a few instances containing named entities, which exhibit weak cohesive ties with the context even if though they are used literally (see Section 2.2.3 for examples). Using a named-entity tagger before applying the GMM might solve the problem.

Finally, Table 5.4 shows the result when using different idioms to generate the nonliteral Gaussian. The literal Gaussian can be generated from the automatically obtained literal examples by the *Boosting the Literal Class* process from Chapter 4. We found the estimation of the GMM is not sensitive to idioms; our model is robust and can use any existing idiom data to discover new figurative expressions (due to shared cohesion structure properties). Furthermore, Table 5.4 also shows that the GMM does not need a large amount of annotated data for parameter estimation. [1] A few hundred instances are sufficient. In our experiments, we find the performance keep stable after 600 instances (randomized) are added to the training set.

5.4 Related Work

There have been many studies on figurative language detection (Birke and Sarkar, 2006; Fazly et al., 2009; Katz and Giesbrecht, 2006; Lin, 1999). Most studies on the detection of figurative

[1] GMM parameter estimation needs two parameters: Gaussian mean and Gaussian covariance. It is relatively simple compared with most supervised machine learning model in which, usually, a large number of parameters need to be estimated. This is why a relatively small number of instances is enough to estimate the corresponding Gaussian component.

Idiom	Class	Precision	Recall	F-Score	Accuracy
bite one's tongue	n	40.79	79.49	53.91	74.94
(166)	l	94.10	73.91	82.79	
break the ice	n	39.05	52.56	44.81	76.12
(541)	l	88.36	81.45	84.77	
pass the buck	n	41.01	73.08	52.53	75.65
(262)	l	92.61	76.23	83.62	

Table 5.4: Results on the figurative expression dataset, Gaussian component parameters estimated on different idioms.

language focus on one of three aspects: type-based extraction (detect idioms on the type level), token-based classification (given a potentially idiomatic phrase in context, decide whether it is used idiomatically), token-based detection (detect figurative expressions which are not lexicalized in running text).

We have disscussed typed-based idiom extraction and token-based idiom classification studies in Chapter 3 and 4. In this chapter, we focus on the third category: token-based detection. There has been relatively little work on token-based detection so far. Fazly et al. (2009) view it as a two stage task which is the combination of type-based extraction and token-based classification. They detect idiom types by using statistical methods that model the general idiomaticity of an expression and then combine this with a simple second-stage process that detects whether the target expression is used figuratively, based on whether the expression occurs in dictionary form.

However, modeling token-based detection as a combination of type-based extraction and token-based classification has some drawbacks. First, type-based approaches typically compute statistics from multiple occurrences of a target expression, hence they cannot be applied to novel usages. Second, these methods were developed to detect figuratively used multi-word expressions (MWEs) and do not work for figuratively used individual words, like *sparrow* in example (2.8). Ideally, one would like to have a generic model that can detect any type of figurative usage in a given context. The model we propose in this chapter is one step in this direction.

5.5 Summary

In this chapter, we aim to not only detect lexicalized figurative expressions but also general unlexicalized figurative expressions. We describe a GMM based approach, which is tested both for distinguishing literal and non-literal usages of a potential idiomatic expression in context and discovering new unlexicalized figurative expressions.

The components of the GMM can be effectively estimated using the EM algorithm. The performance can be further improved when employing an annotated data set for parameter estimation. Our results show that the estimation of Gaussian components are not idiom-dependent. Furthermore, a small annotated data set is enough to obtain good results. In our experiment, we define three types of figurative expressions. Our model works the best for the unambiguous phrase figurative usage (nsu), as this category often violents grammatical rules or selectional constraints, and exhibits the least lexical cohesion with the surrounding context. The ambiguous phrase level figurative usage (nsa) is more difficult as it can be used both literally and figuratively (e.g.: *burn the bridge*). The token level weak figurative usage (nw) is the most difficult category as it is only partially used figuratively. The literally used component words of such expression make the prediction of our GMM model difficult.

One task which we donnot address is detecting the boundaries of a figurative expression, i.e., determining whether it is the whole V+NP expression that is used figuratively or only part of it. We leave this problem to future research.

5. A GAUSSIAN MIXTURE MODEL ON FIGURATIVE EXPRESSION DETECTION

6

Topic Models of Sense Ambiguity

The idiom task discussed in Chapter 3 and 4 and the figurative expression task discussed in Chapters 5 can both be seen as lexical ambiguity problems with two sense categories ('literal' v.s. 'nonliteral'). A more complicated form of sense categories is word sense disambiguation (WSD), where a word may have a number of senses. More concrete examples are discussed in the introduction chapter (e.g., Figure 1.2). The fact that word senses have fuzzier boundaries, compared to the 'literal'/'nonliteral' expression detection task, poses a new challenge for computational modeling. Human annotation agreement studies also suggest that WSD annotation reaches lower agreement than for the idiom 'literal'/'nonliteral' classification task. According to a study conducted by Ng et al. (1999) on WordNet word sense annotation, the average kappa statistic for 5339 instances of 53 nouns is 0.463, whereas the kappa score on the idiom UdSic corpus is 0.700 (Section 2.1).

While being closely related to the idiom and figurative expression detection tasks, WSD poses new challenges in that there exist more candidate sense categories, and the sense boundaries are not as clear (particularly for the fine-grained WSD task).

In this chapter, we aim to deal with these extra challenges by encoding extra human knowledge as probabilistic priors into a Bayesian probabilistic framework. Furthermore, we also aim at developing a uniform framework for both tasks (*WSD* and *'literal'/'nonliteral' expression detection*), as they share the common property that candidate sense categories can be represented by sense paraphrases (since the sense inventory is given). We evaluate our framework on both tasks. We use the UdSic corpus (Section 2.1) for the idiom task and the WSD corpus (Section 2.3) for the word sense disambiguation task.

6.1 Introduction

Word sense disambiguation (WSD) is the task of automatically determining the correct sense for a target word given the context in which it occurs. WSD is an important problem in NLP and an essential preprocessing step for many applications, including machine translation, question answering and information extraction. However, WSD is a difficult task, and despite the fact that it has been the focus of much research over the years, state-of-the-art systems are still often not good enough for real-world applications. One major factor that makes WSD difficult is a relative lack of manually annotated corpora, which hampers the performance of supervised systems.

To address this problem, there has been a significant amount of work on unsupervised WSD that does not require manually sense-disambiguated training data (see McCarthy (2009) for an overview). Recently, several researchers have experimented with topic models (Boyd-Graber and Blei, 2007; Boyd-Graber et al., 2007; Brody and Lapata, 2009; Cai et al., 2007) for sense disambiguation and induction. Topic models are generative probabilistic models of text corpora in which each document is modelled as a mixture over (latent) topics, which are in turn represented by a distribution over words.

Previous approaches using topic models for sense disambiguation either embed topic features in a supervised model (Cai et al., 2007) or rely heavily on the structure of hierarchical lexicons such as WordNet (Boyd-Graber et al., 2007). In this paper, we propose a novel framework which is fairly resource-poor in that it requires only 1) a large unlabelled corpus from which to estimate the topics distributions, and 2) paraphrases for the possible target senses. The paraphrases can be user-supplied or can be taken from existing resources.

We approach the sense disambiguation task by choosing the best sense based on the conditional probability of sense paraphrases given a context. We propose three models which are suitable for different situations: Model I requires knowledge of the prior distribution over senses and directly maximizes the conditional probability of a sense given the context; Model II maximizes this conditional probability by maximizing the cosine value of two topic-document vectors (one for the sense and one for the context). We apply these models to coarse- and fine-grained WSD and find that they outperform comparable systems for both tasks.

We also test our framework on the related task of idiom detection, which involves distinguishing literal and nonliteral usages of potentially ambiguous expressions such as *rock the boat*. For this task, we propose a third model. Model III calculates the probability of a sense given a

context according to the component words of the sense paraphrase. Specifically, it chooses the sense type which maximizes the probability (given the context) of the paraphrase component word with the highest likelihood of occurring in that context. This model also outperforms state-of-the-art systems.

6.2 The Sense Disambiguation Model

6.2.1 Topic Model

As described in PLSA (Hofmann, 1999), the starting point of topic models is to decompose the conditional word-document probability distribution $p(w|d)$ into two different distributions: the word-topic distribution $p(w|z)$, and the topic-document distribution $p(z|d)$ (see Equation 6.1). This allows each semantic topic z to be represented as a multinominal distribution of words $p(w|z)$, and each document d to be represented as a multinominal distribution of semantic topics $p(z|d)$. The model introduces a conditional independence assumption that document d and word w are independent conditioned on the hidden variable, topic z.

$$p(w|d) = \sum_{z} p(z|d)p(w|z) \tag{6.1}$$

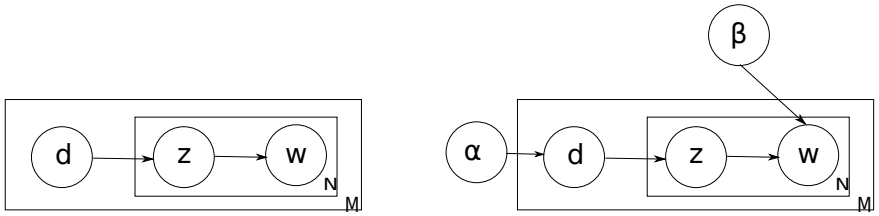
LDA adds Dirichlet hyper-parameters to this framework (Blei et al., 2003). Graphical model representations of PLSA and LDA are represented as Figure 6.1.

The inference of the two distributions given an observed corpus can be done through Gibbs Sampling (Geman and Geman, 1987; Griffiths and Steyvers, 2004). For each turn of the sampling, each word in each document is assigned a semantic topic based on the current word-topic distribution and topic-document distribution. The resulting topic assignments are then used to re-estimate a new word-topic distribution and topic-document distribution for the next turn. This process repeats until a sufficient number of iterations is reached. To avoid statistical coincidence, the final estimation of the distributions is made by the average of the final few rounds.

6.2.2 The Sense Disambiguation Model

Assigning the correct sense s to a target word w occurring in a context c involves finding the sense which maximizes the conditional probability of senses given a context:

$$s = \arg\max_{s_i} p(s_i|c) \tag{6.2}$$



(a) PLSA **(b)** LDA

Figure 6.1: Generative processes of PLSA and LDA. d is document; z is topic; w is word; M is the number of documents in the corpus; N is the number of words within document; α and β are hyper-parameters.

In our models, we represent a sense (s_i) as a collection of 'paraphrases' that capture (some aspect of) the meaning of the sense. These paraphrases can be taken from an existing resource such as WordNet (Miller, 1995) or supplied by the user (see Section 6.3).

This conditional probability is decomposed by incorporating a hidden variable, topic z, introduced by the topic model. We propose three variations of the basic model, depending on how much background information is available (knowledge of the prior sense distribution available and type of sense paraphrases used). In Model I and Model II, the sense paraphrases are obtained from WordNet, and both the context and the sense paraphrases are treated as documents dc, ds.

WordNet is a fairly rich resource which provides detailed information about word senses (glosses, example sentences, synsets, semantic relations between senses, etc.). Sometimes such detailed information may not be available, for instance for languages for which such a resource does not exist or for expressions that are not very well covered in WordNet, such as idioms. For those situations, we propose another model, Model III, in which contexts are treated as documents while sense paraphrases are treated as sequences of independent words.

Model I directly maximizes the conditional probability of the sense given the context, where the sense is modeled as a 'paraphrase document' ds and the context as a 'context document' dc. The conditional probability of the sense given the context $p(ds|dc)$ can be rewritten as a joint probability divided by a normalization factor:

$$p(ds|dc) = \frac{p(ds, dc)}{p(dc)} \tag{6.3}$$

This joint probability can be rewritten as a generative process by introducing a hidden variable z. We make the conditional independence assumption that, conditioned on the topic z,

a paraphrase document ds is generated independently of the specific context document dc:

$$p(ds, dc) = \sum_z p(ds)p(z|ds)p(dc|z) \tag{6.4}$$

We apply the same process to the conditional probability $p(dc|z)$. It can be rewritten as:

$$p(dc|z) = \frac{p(dc)p(z|dc)}{p(z)} \tag{6.5}$$

Now, the disambiguation model $p(ds|dc)$ can be rewritten as a prior $p(ds)$ times a topic function:

$$p(ds|dc) = p(ds) \sum_z \frac{p(z|dc)p(z|ds)}{p(z)} \tag{6.6}$$

We assume $p(z)$ is a uniform distribution, thus $p(z)$ is a constant. Therefore, Equation 6.6 can be rewritten as:

$$p(ds|dc) \propto p(ds) \sum_z p(z|dc)p(z|ds) \tag{6.7}$$

Model I:

$$\arg\max_{ds_i} p(ds_i) \sum_z p(z|dc)p(z|ds_i) \tag{6.8}$$

Model I has the disadvantage that it requires information about the prior distribution of senses $p(ds)$, which is not always available. We use sense frequency information from WordNet to estimate the prior sense distribution, although it must be kept in mind that, depending on the genre of the texts, it is possible that the distribution of senses in the testing corpus may diverge greatly from the WordNet-based estimation. If there is no means for estimating the prior sense distribution of an experimental corpus, generally a uniform distribution is assumed to fulfill the maximum entropy principle (Park and Bera, 2009). However, this assumption does not hold, as the true distribution of word senses is often highly skewed (McCarthy, 2009).

To overcome this problem, we propose Model II, which indirectly maximizes the sense-context probability by maximizing the cosine value of two document vectors that encode the document-topic frequencies from sampling, $v(z|dc)$ and $v(z|ds)$. The document vectors are represented by topics [t_1, t_2, ..., t_n], where t_i represents the number of times that the tokens in this document are assigned to a certain topic.

Model II:

$$\arg\max_{ds_i} \cos(v(z|dc), v(z|ds_i)) \tag{6.9}$$

If the prior distribution of senses is known, Model I is the best choice. However, Model II has to be chosen instead when this knowledge is not available. In our experiments, we test the performance of both models (see Section 6.4).

Sometimes the sense paraphrases are very short, therefore it is difficult to reliably estimate $p(z|ds)$. In order to solve this problem, we treat the sense paraphrase ds as a 'query', a concept which is used in information retrieval. Song and Croft (1999) propose an information retrieval model which takes the conditional probability of the query given the document as a product of all the conditional probabilities of words in the query. The assumption is that the query is generated by a collection of conditionally independent words.

We make the same assumption here. However, instead of taking the product of all the conditional probabilities of words given the document, we take the maximum. There are two reasons for this: (i) taking the product may penalize longer paraphrases since the product of probabilities decreases as there are more words; (ii) we do not want to model the probability of generating specific paraphrases, but rather the probability of generating a sense, which might only be represented by one or two words in the paraphrases (e.g., the potentially idiomatic phrase 'rock the boat' can be paraphrased as 'break the norm' or 'cause trouble'. A similar topic distribution to that of the individual words 'norm' or 'trouble' would be strong supporting evidence of the corresponding idiomatic reading.). We propose,

Model III:

$$\arg\max_{qs_j} \max_{w_i \in qs_j} \sum_z p(w_i|z)p(z|dc) \tag{6.10}$$

where qs is a collection of words contained in the sense paraphrases.

6.2.3 Inference

Here the inference problem boils down to how to estimate the word-topic distribution $p(w|z)$ and topic-document distribution $p(z|d)$. As mentioned earlier, this problem can be done by Gibbs Sampling, which can be described as a two-step iteration process: 1) In the topic assignment step, each word in the document is assigned a semantic topic. The probability of a word being assigned to a topic conditioned on other variables $p(z_j|w_i, d, z_{-j}, w_{-i})$ is estimated by the product of the probability of generating a topic given a document $p(z_j|d)$, and the probability of

generating a word given a topic $p(w_i|z_j)$. 2) In the Estimation step, the topic assignments from the first stage are used to re-estimate the topic-word distribution and word-topic distribution, which, in turn, change the topic assignments in the next iteration.

Sampling (Gibbs):

$$p(z_j|w_i, d, z_{-j}, w_{-i}) \propto p(w_i|z_j) * p(z_j|d) \tag{6.11}$$

Probability Estimation:

$$p(z_j|d) = \frac{\sum_{w_k \in d} f(w_k, z_j) + \alpha}{\sum_{d_n} \sum_{w_m \in d_n} f(w_m, z_j) + T\alpha} \tag{6.12}$$

$$p(w_i|z_j) = \frac{f(w_i, z_j) + \beta}{\sum_{w_k} f(w_k, z_j) + W\beta} \tag{6.13}$$

where $p(z_j|w_i, d, z_{-j}, w_{-i})$ is the the i^{th} word w_i in document d is assigned with topic z_j; $p(w_i|z_j)$ is the probability of word w_i given topic z_j; $p(z_j|d)$ is the probability of a topic z_j given a document d; $f(w_k, z_j)$ is the number of times that word w_k is assigned with topic z_j; α and β are Dirichlet hyper-parameters; T is the topic size; W is the vocabulary size.

In our experiment, one possible way to run inference is to combine the context documents and sense paraphrases into a corpus and run Gibbs sampling on top of this. The problem with this approach is that the test set and sense paraphrase set are relatively small, and topic models running on a small corpus are less likely to capture rich semantic topics. One simple explanation is that a small corpus usually has a relatively small vocabulary, which is less representative of topics, i.e., $p(w|z)$ cannot be estimated reliably.

In order to overcome this problem, we infer the word-topic distribution from a very large corpus (Wikipedia dump, see Section 6.3). All of the following inference experiments on the test corpus are based on the assumption that the word-topic distribution $p(w|z)$ is the same as the one estimated from the Wikipedia dump. Inference of topic-document distributions for context and sense paraphrases is done by fixing the word-topic distribution as a constant.

6.3 Experimental Setup

We evaluate our models on three different tasks: coarse-grained WSD, fine-grained WSD and literal vs. nonliteral sense detection. In this section we discuss our experimental set-up, and

POS	Paraphrase reference synsets
N	hyponyms, instance hyponyms, member holonyms, substance holonyms, part holonyms, member meronyms, part meronyms, substance meronyms, attributes, topic members, region members, usage members, topics, regions, usages
V	Troponyms, entailments, outcomes, phrases, verb groups, topics, regions, usages, sentence frames
A	similar, pertainym, attributes, related, topics, regions, usages
R	pertainyms, topics, regions, usages

Table 6.1: Selected reference synsets from WordNet that were used for different parts-of-speech to obtain word sense paraphrase. N(noun), V(verb), A(adj), R(adv).

how we choose sense paraphrases and instance contexts.

Sense Paraphrases For word sense disambiguation tasks, the paraphrases of the sense keys are represented by information from WordNet 2.1. (Miller, 1995). To obtain the paraphrases, we use the *word forms*, *glosses* and *example sentences* of the synset itself and a set of selected *reference synsets* (i.e., synsets linked to the target synset by specific semantic relations, see Table 6.1).[1] For instance, the *Instance Hyponyms* reference synset is defined as "specific (usually real-word) instance of this type". An example "Instance Hyponyms" of the word *river* is *Mississippi river*. We excluded the 'hypernym reference synsets', since information common to all of the child synsets may confuse the disambiguation process.

For the literal vs. nonliteral sense detection task, we selected the paraphrases of the nonliteral meaning from several online idiom dictionaries. For the literal senses, we used 2-3 manually selected words with which we tried to capture (aspects of) the literal meaning of the expression.[2] For instance, the literal paraphrases that we chose for 'break the ice' were *ice, water* and *snow*.[3] The paraphrases are shorter for the idiom task than for the WSD task, because the meaning descriptions from the idiom dictionaries are shorter than what we get from WordNet. In the latter case, each sense can be represented by its synset as well as its reference synsets.

[1] An example of the word sense paraphrases is in the Appendix A.1).

[2] Note that we use the word 'paraphrase' in a fairly wide sense in this paper. Sometimes it is not possible to obtain exact paraphrases. This applies especially to the task of distinguishing literal from nonliteral senses of multi-word expressions. In this case we take as paraphrases some key words which capture salient aspects of the meaning.

[3] The complete sense paraphrases of the idioms are represented in the appendix (Table A.2).

Instance Context We experimented with different context sizes for the disambiguation task. The five different context settings that we used for the WSD tasks are: collocations (1w), ±5-word window (5w), ±10-word window (10w), current sentence, and whole text. Because the idiom corpus also includes explicitly marked paragraph boundaries, we included 'paragraph' as a sixth type of context size for the idiom sense detection task.

6.4 Experiments

As mentioned above, we test our proposed sense disambiguation framework on three tasks. We start by describing the sampling experiments for estimating the word-topic distribution from the Wikipedia dump. We used the package provided by Wang et al. (2009) with the suggested Dirichlet hyper-parameters.[1] In order to avoid statistical instability, the final result is averaged over the last 50 iterations. We did four rounds of sampling with 1000, 500, 250, and 125 topics respectively. The final word-topic distribution is a normalized concatenate of the four distributions estimated in each round. In average, the sampling program run on the Wikipedia dump consumed 20G memory, and each round took about one week on a single AMD Dual-Core 1000MHZ processor.

6.4.1 Coarse-Grained WSD

In this section we first describe the landscape of similar systems against which we compare our models, then present the results of the comparison. The systems that participated in the SemEval-2007 coarse-grained WSD task (Task-07) can be divided into three categories, depending on whether training data is needed and whether other types of background knowledge are required: What we call Type I includes all the systems that need annotated training data. All the participating systems that have the mark *TR* fall into this category (see Navigli et al. (2007) for the marks *TR,MFS*). Type II consists of systems that do not need training data but require prior knowledge of the sense distribution (estimated sense frequency). All the participating systems that have the mark *MFS* belong to this category. Systems that need neither training data nor prior sense distribution knowledge are categorized as Type III.

We make this distinction based on two principles: (i) the cost of building a system; (ii) the portability of the established resource. Type III is the cheapest type of system to build, while Type I and Type II both need extra resources. Type II has an advantage over Type I

[1] They were set as: $\alpha = \frac{50}{\#topics}$ and $\beta = 0.01$.

System	Noun	Verb	Adj	Adv	All
UoR-SSI	84.12	78.34	85.36	88.46	83.21
NUS-PT	82.31	78.51	85.64	89.42	82.50
UPV-WSD	79.33	72.76	84.53	81.52	**78.63***
TKB-UO	70.76	62.61	78.73	74.04	**70.21**$'$
MII–ref	78.16	70.39	79.56	81.25	76.64
MII+ref	80.05	70.73	82.04	82.21	**78.14**$'$
MI+ref	79.96	75.47	83.98	86.06	**79.99***
BL$_{mfs}$	77.44	75.30	84.25	87.50	**78.99***

Table 6.2: Model performance (F-score) on the coarse-grained dataset (context=sentence). Performance on different part-of-speech tags. For our model, (+ref/-ref) indicates whether we use reference synsets.

since the prior knowledge of the sense distribution can be estimated from annotated corpora (e.g.: SemCor, Senseval). In contrast, training data in Type I may be system specific (e.g.: different input format, different annotation guidelines). McCarthy (2009) also addresses the issue of performance and cost by comparing supervised word sense disambiguation systems with unsupervised ones.

We exclude the system provided by one of the organizers (UoR-SSI) from our categorization. The reason is that although this system is claimed to be unsupervised, and it performs better than all the participating systems (including the supervised systems) in the SemEval-2007 shared task, it still needs to incorporate a lot of prior knowledge, specifically information about co-occurrences between different word senses, which was obtained from a number of resources (SSI+LKB) including: (i) SemCor (manually annotated); (ii) LDC-DSO (partly manually annotated); (iii) collocation dictionaries which are then disambiguated semi-automatically. Even though the system is not "trained", it needs a lot of information which is largely dependent on manually annotated data, so it does not fit neatly into the categories Type II or Type III either.

Table 6.2 lists the best participating systems of each type in the SemEval-2007 task (Type I: NUS-PT (Chan et al., 2007); Type II: UPV-WSD (Buscaldi and Rosso, 2007); Type III: TKB-UO (Anaya-Sánchez et al., 2007)). Our Model I belongs to Type II, and our Model II belongs to Type III.

Table 6.2 compares the performance of our models with the Semeval-2007 participating

systems. We only compare the F-score, since all the compared systems have an attempted rate[1] of 1.0, which makes both the precision and recall rates the same as the F-score. We focus on comparisons between our models and the best SemEval-2007 participating systems within the same type. Model I is compared with UPV-WSD, and Model II is compared with TKB-UO. In addition, we also compare our system with the most frequent sense baseline which was not outperformed by any of the systems of Type II and Type III in the SemEval-2007 task.

Comparison on Type III is marked with $'$, while comparison on Type II is marked with $*$. We find that Model II performs statistically significantly better than the best participating system of the same type TKB-UO ($p<<0.01$, χ^2 test). When encoded with the prior knowledge of sense distribution, Model I outperforms by 1.36% the best Type II system UPV-WSD, although the difference is not statistically significant. Furthermore, Model I also quantitatively outperforms the most frequent sense baseline BL_{mfs}, which, as mentioned above, was not beat by any participating systems that do not use training data.

We also find that our model works best for nouns. The unsupervised Type III model Model II achieves better results than the most frequent sense baseline on nouns, but not on other parts-of-speech. This is in line with results obtained by previous systems (Boyd-Graber and Blei, 2008; Cai et al., 2007; Griffiths et al., 2005). While the performance on verbs can be increased to outperform the most frequent sense baseline by including the prior sense probability, the performance on adjectives and adverbs remains below the most frequent sense baseline. We think that there are three reasons for this: first, adjectives and adverbs have fewer reference synsets for paraphrases compared with nouns and verbs (see Table 6.1); second, adjectives and adverbs tend to convey less key semantic content in the document, so they are more difficult to capture with the topic model; and third, adjectives and adverbs are a small portion of the test set, so their performances are statistically unstable. For example, if 'already' appears 10 times out of 20 adverb instances, a system may get bad results on adverbs only because of its failure to disambiguate the word 'already'.

Paraphrase analysis Table 6.2 also shows the effect of different ways of choosing sense paraphrases. MII+ref is the result of including the reference synsets, while MII-ref excludes the reference synsets. As can be seen from the table, including all reference synsets in sense

[1] Attempted rate is defined as the total number of disambiguated output instances divided by the total number of input instances.

Context	Ate.	Pre.	Rec.	F1
$\pm 1w$	91.67	75.05	68.80	71.79
$\pm 5w$	99.29	77.14	76.60	76.87
$\pm 10w$	100	77.92	77.92	77.92
text	100	76.86	76.86	76.86
sent.	100	**78.14**	**78.14**	**78.14**

Table 6.3: Model II performance on different context sizes. attempted rate (Ate.), precision (Pre.), recall (Rec.), F-score (F1).

System	F-score
RACAI	52.7 $\pm$4.5
BL_{mfs}	55.91$\pm$4.5
MI+ref	**56.99**$\pm$4.5

Table 6.4: Model performance (F-score) for the fine-grained word sense disambiguation task.

paraphrases increases performance. Longer paraphrases contain more information, and they are statistically more stable for inference.

We find that nouns get the greatest performance boost from including reference synsets, as they have the largest number of different types of synsets. We also find the 'similar' reference synset for adjectives to be very useful. Performance on adjectives increases by 2.75% when including this reference synset.

Context analysis In order to study how the context influences the performance, we experiment with Model II on different context sizes (see Table 6.3). We find *sentence context* is the best size for this disambiguation task. Using a smaller context not only reduces the precision, but also reduces the recall rate, which is caused by the all-zero topic assignment by the topic model for documents only containing words that are not in the vocabulary. As a result, the model is unable to disambiguate. The context based on the whole text (article) does not perform well either, possibly because using the full text folds in too much noisy information.

6.4.2 Fine-grained WSD

We saw in the previous section that our framework performs well on coarse-grained WSD. Fine-grained WSD, however, is a more difficult task. To determine whether our framework is also able to detect subtler sense distinctions, we tested Model I on the English all-words subtask of SemEval-2007 Task-17 (see Table 6.4).

We find that Model I performs better than both the best unsupervised system, RACAI (Ion and Tufiş, 2007) and the most frequent sense baseline (BL_{mfs}), although these differences are not statistically significant due to the small size of the available test data (465).

6.4.3 Idiom Sense Disambiguation

In the previous section, we provided the results of applying our framework to coarse- and fine-grained word sense disambiguation tasks. For both tasks, our models outperform the state-of-the-art systems of the same type either quantitatively or statistically significantly. In this section, we apply Model III to another sense disambiguation task, namely distinguishing literal and nonliteral senses of ambiguous expressions.

WordNet has a relatively low coverage for idiomatic expressions. In order to represent non-literal senses, we replace the paraphrases obtained automatically from WordNet by words selected manually from online idiom dictionaries (for the nonliteral sense) and by linguistic introspection (for the literal sense). We then compare the topic distributions of literal and nonliteral senses.

As the paraphrases obtained from the idiom dictionary are very short, we treat the paraphrase as a sequence of independent words instead of as a document and apply Model III (see Section 6.2). Table 6.5 shows the results of our proposed model compared with state-of-the-art systems. We find that the system significantly outperforms the majority baseline ($p<<0.01$, χ^2 test) and the cohesion-graph based approach proposed by Sporleder and Li (2009) ($p<<0.01$, χ^2 test). The system also outperforms the bootstrapping system by Li and Sporleder (2009), although not statistically significantly. This shows how a limited amount of human knowledge (e.g., paraphrases) can be added to an unsupervised system for a strong boost in performance (Model III compared with the cohesion-graph and the bootstrapping approaches).

For obvious reasons, this approach is sensitive to the quality of the paraphrases. The paraphrases chosen to characterise (aspects of) the meaning of a sense should be non-ambiguous between the literal or idiomatic meaning. For instance, ‘fire’ is not a good choice for a paraphrase

System	Prec_l	Rec_l	F_l	Acc.
Base_{maj}	-	-	-	78.25
co-graph	50.04	69.72	58.26	78.38
boot.	71.86	66.36	69.00	87.03
Model III	67.05	81.07	73.40	87.24

Table 6.5: Performance on the literal or nonliteral sense disambiguation task on idioms. literal precision (Prec_l), literal recall (Rec_l), literal F-score (F_l), accuracy(Acc.).

of the literal reading of 'play with fire', since this word can be interpreted literally as 'fire' or metaphorically as 'something dangerous'. The verb component word 'play' is a better literal paraphrase.

For the same reason, this approach works well for expressions of which the literal and nonliteral readings are well separated (i.e., occur in different contexts), while the performance drops for expressions whose literal and idiomatic readings can appear in a similar context. We test the performance on individual idioms on the five most frequent idioms in our corpus[1] (see Table 6.6). We find that 'drop the ball' is a difficult case. The words 'fault', 'mistake', 'fail' or 'miss' can be used as the nonliteral paraphrases. However, it is also highly likely that these words are used to describe a scenario in a baseball game, in which 'drop the ball' is used literally. In contrast, the performance on 'rock the boat' is much better, since the nonliteral reading of the phrases 'break the norm' or 'cause trouble' are less likely to be linked with the literal reading 'boat'. This may also be because 'boat' is not often used metaphorically in the corpus.

As the topic distribution of nouns and verbs exhibit different properties, topic comparisons across parts-of-speech do not make sense. We make the topic distributions comparable by making sure each type of paraphrase contains the same sets of parts-of-speech. For instance, we do not permit combinations of literal paraphrases which only consist of nouns and nonliteral paraphrases which only consist of verbs.

6.5 Related Work

There is a large body of work on WSD, covering supervised, unsupervised (word sense induction) and knowledge-based approaches (see McCarthy (2009) for an overview). While most

[1] We tested only on the most frequent idioms in order to avoid statistically unreliable observations.

Idiom	Acc.
drop the ball	75.86
play with fire	91.17
break the ice	87.43
rock the boat	95.82
set in stone	89.39

Table 6.6: Performance on individual idioms.

supervised approaches treat the task as a classification task and use hand-labelled corpora as training data, most unsupervised systems automatically group word tokens into similar groups using clustering algorithms, and then assign labels to each sense cluster. Knowledge-based approaches exploit information contained in existing resources. They can be combined with supervised machine-learning models to assemble semi-supervised approaches.

Recently, a number of systems have been proposed that make use of topic models for sense disambiguation. Cai et al. (2007), for example, use LDA to capture global context. They compute topic models from a large unlabelled corpus and include them as features in a supervised system. Boyd-Graber and Blei (2007) propose an unsupervised approach that integrates McCarthy et al. (2004) method for finding predominant word senses into a topic modelling framework. In addition to generating a topic from the document's topic distribution and sampling a word from that topic, the enhanced model also generates a distributional neighbour for the chosen word and then assigns a sense based on the word, its neighbour and the topic. Boyd-Graber and Blei (2007) test their method on WSD and information retrieval tasks and find that it can lead to modest improvements over state-of-the-art results.

In another unsupervised system, Boyd-Graber et al. (2007) enhance the basic LDA algorithm by incorporating WordNet senses as an additional latent variable. Instead of generating words directly from a topic, each topic is associated with a random walk through the WordNet hierarchy which generates the observed word. Topics and synsets are then inferred together. While Boyd-Graber et al. (2007) show that this method can lead to improvements in accuracy, they also find that idiosyncracies in the hierarchical structure of WordNet can harm performance. This is a general problem for methods which use hierarchical lexicons to model semantic distance (Budanitsky and Hirst, 2006). In our approach, we circumvent this problem by exploiting paraphrase information for the target senses rather than relying on the structure of WordNet as a

whole.

Topic models have also been applied to the related task of word sense induction. Brody and Lapata (2009) propose a method that integrates a number of different linguistic features into a single generative model. Topic models have also been previously considered for metaphor extraction and estimating the frequency of metaphors (Bethard et al., 2009; Klebanov et al., 2009) (related to our experiments tested on the idiom dataset).

6.6 Summary

In this chapter, we model sense disambiguation on words (multiple sense categories problem) and idiomatic expressions (binary 'literal'/'nonliteral' problem) in one uniform topic model framework. Consequently, we propose three sub-models. The basic idea of these models is to compare the topic distribution of a target instance with the candidate sense paraphrases and choose the most probable one. While Model I and Model III model the problem in a probabilistic way, Model II uses a vector space model by comparing the cosine values of two topic vectors. Model II and Model III are completely unsupervised, while Model I needs the prior sense distribution. Model I and Model II treat the sense paraphrases as documents, while Model III treats the sense paraphrases as a collection of independent words.

We test the proposed models on three tasks. We apply Model I and Model II to the WSD tasks due to the availability of more paraphrase information. Model III is applied to the idiom detection task since the paraphrases from the idiom dictionary are smaller. We find that all models outperform comparable state-of-the-art systems either quantitatively or statistically significantly.

By testing our framework on three different sense disambiguation tasks, we show that the framework can be used flexibly in different application tasks. The system also points out a promising way of solving the granularity problem of word sense disambiguation, as new application tasks which need different sense granularities can utilize this framework when new paraphrases of sense clusters are available. In addition, this system can also be used in a larger context such as sentiment detection (*positive* or *negative*).

7

From Disambiguation to Induction: the Evaluation Bottleneck

We have discussed various lexical ambiguity problems in the previous chapters. We started with the idiom detection task in which candidate senses are constrained to'literal' and 'idiomatic' (Chapter 3 and 4). Then in Chapter 5 we moved on to a more general case in which target phrases are not lexicalized but they bear special semantic meanings (e.g., metaphor, metonymy). We define it as a general figurative expression detection task which, similar to the idiom task, has two candidate sense categories ('literal' and 'nonliteral'). In Chapter 6, we considered a more complicated situation in which words often have more than two candidate senses and the sense boundaries are fuzzier, which is known as WSD. In this chapter we discuss another lexical ambiguity problem in which the sense inventory is not explicitly specified, known as word sense induction (WSI).

WSI research is hindered by deficiencies of commonly used evaluation measures. As shown in the SemEval 2010 WSI shared task, simple baselines tend to win over state-of-the-art systems (e.g., the most-frequent-sense baseline wins under the paired F-Score; the one-cluster-per-instance wins under the V-Measure). With this in mind, in this chapter we focus on improved evaluation measures which would eliminate the obstacles of WSI research and bring promising research perspectives to this topic.

We first give an introduction to WSI and discuss why WSI evaluation is a difficult task. Then, we give an overview of different evaluation approaches and list some well-known problems of those evaluation approaches. In Section 7.3 and 7.4, we discuss two new findings from our research:

1. The state-of-the-art supervised evaluation approach has problems: backoff to the most-frequent-baseline, bias towards fine-grained output, and sensitivity to training-test split.

2. The entropy estimation of the state-of-the-art unsupervised evaluation (V-Measure) is biased, and the bias can be alleviated by using alternative entropy estimators.

Furthermore, we also demonstrate that WSI evaluation can be more effective if the task setup constrains systems to output a limited number of clusters based on test set size and the number of gold standard senses. The reason is that entropy bias tends to be sensitive to the ratio of cluster number to class number and sample (instance) size. We also show the bias can be significantly reduced if the number of clusters is restricted to be less than a certain threshold defined by number of gold classes and the number of instances. At the end of this chapter, we provide a concrete strategy for restricting the number of clusters.

7.1 Introduction

Word sense induction (WSI) differs from word sense disambiguation (WSD) in that induction systems are not equipped with knowledge of a sense inventory. Senses have to be induced automatically in an unsupervised fashion on the basis of corpus evidence (Purandare and Pedersen, 2004; Schütze, 1998). The independence of a fixed sense inventory is one of the main advantages of WSI. Fixed inventories assume some kind of ground truth of senses for a given target word. However, lexicographers find it typically difficult to agree on a fixed sense inventory or even on the ideal level of sense granularity. In fact, inventories of different granularities may be necessary for different domains and applications. Moreover, some researchers argue that graded representations of word meaning better reflect linguistic realities than rigid sense inventories (Cruse, 2000). Since WSI approaches induce their own sense inventories they can automatically adapt to the target domain. They may thus be beneficial for applications such as information retrieval and machine translation, which have been shown to benefit from induced senses (Véronis, 2004; Vickrey et al., 2005).

However, the fact that WSI systems do not rely on fixed inventories also makes it notoriously difficult to evaluate and compare the performance of different approaches. Ideally, it should be possible to compare systems even if the induced inventories are of different granularity. However, most existing evaluation methods tend to be biased towards either very coarse-grained or very fine-grained clusterings. In this chapter, we look at different evaluation approaches that

have been used for WSI shared tasks (Agirre and Soroa, 2007; Manandhar et al., 2010), discuss their weaknesses and outline alternative solutions for future research.

7.2 Overview of Evaluation Approaches

There exist two main evaluation schemes, both of which assume ground truth in the form of a fixed inventory of gold standard senses and evaluate the system by comparing the set of induced senses (*clusters*) to the set of gold standard sense annotations in the test data (*classes*). The first scheme, called *unsupervised evaluation*, employs standard cluster evaluation methods. The induced senses, which are represented as clusters of usages of the target word, are compared to sets of examples in the gold standard and then evaluated using measures which assess the quality of the clustering such as *F-Score* (Agirre and Soroa, 2007), *paired F-Score* (Manandhar et al., 2010), *Entropy* (Zhao and Karypis, 2005), *Purity* (Zhao and Karypis, 2005), or normalized mutual information also known as *V-Measure* (Rosenberg and Hirschberg, 2007; Strehl and Gosh, 2002). While unsupervised evaluation directly compares two different partitions of target word uses (produced by the WSI system on the one hand and the manual annotation on the other hand), supervised evaluation (Agirre and Soroa, 2007) uses annotated data to map the induced sense clusters to gold standard sense classes and then uses this mapping to tag the test data. The tagging produced can then be evaluated using standard supervised evaluation measures such as accuracy. To compute the mapping, the evaluation data are split into a *mapping set*, which is used to determine the best mapping of clusters to classes, and a *test set*, which is subsequently tagged according to this mapping.

F-Score Measure (Agirre and Soroa, 2007) defines the F-Score of a sense class c_r as the maximum F-Score attained at any cluster k_i by mapping[1]:

$$F(c_r) = \max_{k_i} f(c_r, k_i) \tag{7.1}$$

This strategy allows each sense class to be mapped to any cluster which suggests one cluster may be mapped to multiple sense classes. This many-to-one mapping from classes to clusters is biased towards coarse-grained output (e.g., the most-frequent-sense baseline). As examples, the most-frequent-sense baseline achieves the best performances in both SemEval 2007 and SemEval 2010 shared tasks under F-Score evaluation.

[1] In this chapter, we use c to denote gold classes and k to denote system output clusters.

F-Score measure restricts the maximum number of mapped clusters to be the number of gold sense classes. Thus, a system would be penalized if it outputs more clusters than gold classes, as extra clusters would not have a chance to be mapped to the gold sense classes, i.e, the F-Score measure penalizes fine-grained output (Rosenberg and Hirschberg, 2007). As an example, the best supervised system of SemEval10, UoY, which generates an average of 11.54 clusters is only ranked 16th by F-Score evaluation.

Entropy Measure (Zhao and Karypis, 2005) considers how instances from gold sense classes are distributed within each cluster. This approach measures the entropy of the probability of sense classes conditioned on clusters $p(c|k)$: the lower the entropy is, the better the system is ranked. This approach evaluates how an induced cluster entails a gold sense class but ignores how a gold sense class entails a cluster. Therefore, it allows many-to-one mapping from clusters to classes, which in turn encourages fine-grained output. One example is the one-cluster-per-instance baseline, which achieves the best results in the SemEval 2007 shared task under the Entropy measure (entropy 0).

Purity Measure (Zhao and Karypis, 2005) considers the extent to which each cluster contains objects from primarily one class: a larger purity value means better clustering algorithm. Exhibiting an opposite problem of the Entropy measure, it only measures how cluster entails class, and consequently, encourages fine-grained output by allowing one-to-many mapping from classes to clusters (the best system of the Semeval 2007 WSI task is the one-instance-per-cluster baseline with a purity of 100%).

The problems of the *F-Score*, *Entropy* and *Purity* evaluation measures have been addressed in the most recent SemEval shared task: SemEval 2010 abandoned the *Entropy* and *Purity* measures, and replaced the *F-Score* measure by an improved version *paired F-Score* (Artiles et al., 2009). However, as far as we know there have been no studies on the effectiveness of the two remaining approaches that were also adopted by SemEval 2010 (*supervised evaluation* and *V-Measure*). In the next two sections, we show our new findings concerning the problems of these two approaches.

7.3 Finding 1: The State-of-the-Art Supervised Evaluation Favors Fine-grained Output

We start this section by discussing three phenomena from the SemEval 2010 shared task evaluation: 1) The supervised evaluation seems to compress the results of all the systems into a narrow band that converges around the most-frequent-sense result (Pedersen, 2010). 2) The best system under the supervised evaluation measure (UoY (Korkontzelos and Manandhar, 2010)) generates an average number of 11.54 clusters, which is twice as many as the gold standard annotation (average 5.6 sense clusters).[1] 3) The ranking inconsistency behavior: The 80-20 training and test split (80% for training, and 20% for testing) evaluation strategy does not completely comply with the 60-40 split under the supervised evaluation.

In order to illustrate this evaluation issue, we first give a brief introduction to the supervised evaluation approach proposed by Agirre et al. (2006). In this approach, a mapping matrix $(M)_{ij} = p(c_j|k_i)$ that relates hubs k_i (induced clusters) and senses c_j (gold standard class) is built based on the conditional probability of a word having sense j given that it has been assigned hub i. This conditional probability can be calculated by the counts of sense c_j being assigned hub k_i in the training set. For example, if a co-occurrence matrix is as the left side of Example 7.2, then the mapping matrix M would be generated by normalizing the row vector, resulting in the matrix on the right-hand side of Formula 7.2.

$$\begin{bmatrix} & c_1 & c_2 \\ k_1 & 1 & 0 \\ k_2 & 1 & 1 \\ k_3 & 0 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} & c_1 & c_2 \\ k_1 & 1 & 0 \\ k_2 & 0.5 & 0.5 \\ k_3 & 0 & 1 \end{bmatrix} \tag{7.2}$$

Given a test instance with the hub assignment as $k = (k_1, ..., k_m)$, the sense score vector is calculated by multiplying the hub vector by M, i.e., $c = k' \times M$. The sense class of the test instance is decided by the maximum sense score. In our study we find three problems with this proposed approach:

1. The evaluation model backs off to the most-frequent-sense baseline. Suppose that we have a naive system which always outputs 3 hubs with equal weights to any input instance. Consider the case that there are 4 instances in the training set: 3 of c_1, 1 of c_2. The co-occurance

[1] Furthermore, the UoY system has a tendency to output a large number of equally weighted classes for each instance. For example, the system assigns 351 classes with equal weights to 11 instances (out of 31) of the noun 'accounting' in the test set.

matrix which is based on the fraction counts would be as shown on the left side of Example 7.3. The normalized mapping matrix M is as the right side of Formula 7.3. Now if we have a new test instance with the hub assignment $k = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, the sense score vector would be $c = k' \times M = (\frac{3}{4}, \frac{1}{4})$. Thus, the system would output c_1 as the mapped sense class. This naive system gains performance by exploiting the distribution information in the training set.

$$\begin{bmatrix} & c_1 & c_2 \\ k_1 & 1 & \frac{1}{3} \\ k_2 & 1 & \frac{1}{3} \\ k_3 & 1 & \frac{1}{3} \end{bmatrix} \rightarrow \begin{bmatrix} & c_1 & c_2 \\ k_1 & \frac{3}{4} & \frac{1}{4} \\ k_2 & \frac{3}{4} & \frac{1}{4} \\ k_3 & \frac{3}{4} & \frac{1}{4} \end{bmatrix} \tag{7.3}$$

In general, any system would benefit from backing off to the distributions in the evaluation training data. Actually, the sense score calculator ($c = k' \times M$) is eventually a process of mixing two distributions: the system output hub distribution (represented by hub score k') and the sense distribution in the training data (represented by the mapping matrix M). The consequence is that system output is conflated with the most-frequent-sense (estimated from the training data) baseline. This also explains why all the system performances fall into a narrow band that converges around the most-frequent-sense result (Pedersen, 2007).

2. The evaluation is biased towards fine-grained output. Suppose we have a training set which consists of 3 instances with the hub assignments as: $k(I_1)$=(1, 0, 0), $k(I_2)$=(0, 1, 0), $k(I_3)$=(0, 0, 1). The gold senses of these three examples are as $I_1 \in c_1$, and $I_2, I_3 \in c_2$. The mapping matrix based on this training set is as:

$$M = \begin{bmatrix} & c_1 & c_2 \\ k_1 & 1 & 0 \\ k_2 & 0 & 1 \\ k_3 & 0 & 1 \end{bmatrix}$$

Now we have test instance with a hub output as $k(I_t)$=(0.4, 0.3, 0.3), and gold sense label as $I_t \in c_2$. We consider two cases:

- The system makes an early decision. It selects the hub with the highest weight as its final output $I_t \in k_1$. The output hub score would be $k(I_t)$=(1, 0, 0), and the final sense score would be $c = k(I_t)' \times M = (1, 0)$. As c_1 has a higher sense score, the evaluation system would then choose c_1 as the mapped sense.

- The system makes a lazy decision, instead of selecting a single class at an early stage, it presents the whole weighted induced cluster vector (0.4, 0.3, 0.3) to the evaluation model, which would, in turn, use the mapping matrix estimated from the training data to map the weighted hub output to a final sense output $c = k' \times M$ =(0.4, 0.6). In this case, the system gets the instance correct (output mapped sense class as c_2) by the backoff knowledge in the mapping matrix.

Another view on this: the training phrase of the evaluation model can be seen as a feature selection process, in which the most informative system output hubs are correlated to gold sense classes. The system has the freedom to select the important hubs from the training data. If a system only assigns one hub for each instance, it is the same as submitting a single feature property to the evaluation model, which would have disadvantages compared to systems that submit more properties (features). In the extreme case, one can submit a bigram co-ocurrence matrix of the target word as the system hub output and let the evaluation model correlate this output to the gold sense class by the mapping matrix. If the evaluation model has enough trainning data, it can effectively learn the correlations and make predictions. What we compare in such a setting is actually *feature selection*. Systems making decisions at an early stage lose the advantage of utilizing the training data in the evaluation stage to tune the most powerful features. As a result, this evaluation measure penalizes under-generation (e.g., only output one most confident cluster), but encourages over-generation (e.g., output a large number of equally weighted clusters).

3. Inconsistent behavior with respect to the training-test split. As in most machine learning applications, model quality increases with the size of the training set. This explains why the 80-20 training-test split gains better results than 60-40 split for all the top 10 systems. However, the fact that the rankings produced by the two strategies are inconsistent, is another disadvantage of the supervised evaluation. For instance, the system Duluth-WSI is ranked as the second in the 80-20 split while it is only ranked as the fifth in the 60-40 split in SemEval 2010 WSI task (Manandhar et al., 2010).

7.4 Finding 2: The Entropy Bias Problem of the V-Measure

In this section, we discuss the entropy estimator bias problem of the V-Measure WSI evaluation approach by comparing different estimators and their influence on the evaluation scores. We

also run two simulation experiments using uniform and Zipf distributions where we know the true entropy. Our experiments show that standard Maximum Likelihood Estimator (MLE) is very inaccurate in case of WSI evaluation where the number of clusters m is comparable to the number of samples N. We also find that there exist better estimators and argue that they should replace MLE for WSI evaluation.

7.4.1 Normalized Mutual Information (V-Measure)

Various efforts have been made to improve WSI evaluation. Information theoretic based measures are one major type. The basic idea with these methods is to take the sense class and system cluster as two random variables, and the evaluation measure evaluates how those two random variables are related to each other. This type of evaluation methodology has been adopted in general clustering algorithm evaluation (e.g. (Dom, 2001; Meila, 2007; Strehl and Gosh, 2002)). However, specific usage in the WSI evaluation has only recently been adopted by SemEval 2010 (Manandhar et al., 2010), in which Normalized Mutual Information ((Strehl and Gosh, 2002)), also called V-Measure (Rosenberg and Hirschberg, 2007), is utilized.

The starting point of using information theoretic measures is to check how two random variables, class c and class k, depend on each other. One natural choice is to use the Mutual Information (MI),

$$I(c,k) = H(c) + H(k) - H(c,k) \tag{7.4}$$

$$= H(c) - H(c|k) \tag{7.5}$$

$$= H(k) - H(k|c) \tag{7.6}$$

where H is defined in terms of Shannon's entropy (Shannon and Weaver, 1998). If p denotes the probability mass function, then entropy is calculated as:

$$H(x) = -\sum_{i=1}^{m} p(x_i) \log p(x_i) \tag{7.7}$$

In case of $p(x_i) = 0$ for some i, the corresponding value is set to be 0, as $\lim_{p \to 0+} p \log p = 0$. In the MI definition, $H(c)$, $H(k)$ is the marginal variable entropy, $H(c,k)$ is the joint entropy of two variables, and $H(c|k)$, $H(k|c)$ is the conditional entropy.

Different Variations of such measures have also been proposed such as Variation of Information Measure (Meila, 2007), $VI(c,k) = H(c|k) + H(k|c)$, and Q_0 Measure (Dom, 2001), $H(c|k)$. A common problem of these evaluation approaches is that they are not normalized,

which makes the comparison of scores from different systems difficult. For instance, a system which outputs a large number of clusters may have a larger MI value, as they may have a larger marginal entropy compared with systems which output a small number of clusters.

Normalized Mutual Information (Rosenberg and Hirschberg, 2007; Strehl and Gosh, 2002) aims to solve the above issue by using the harmonic mean of two normalized entropy parameters:

$$V(c,k) = \frac{1}{\frac{H(c)}{I(c,k)} + \frac{H(k)}{I(c,k)}} = \frac{2I(c,k)}{H(c)+H(k)} \tag{7.8}$$

where $\frac{I(c,k)}{H(c)}$, $\frac{I(c,k)}{H(k)}$ is MI normalized by marginal entropy.

Our study on the V-Measure was motivated by the fact that the one-cluster-per-instance baseline (1cl1inst) [1] in SemEval 2010 yields a V-Score of 31.7% which is significantly higher than the top participating system (Hermit 16.2%). In our subsequent study, we find that the V-Measure is positively biased as a result of the fact that the state-of-the-art WSI entropy estimator, Maximum Likelihood Estimator (MLE), is heavily negatively biased when the sample size is relatively small compared with the number of clusters (the case for WSI tasks such as SemEval 2010). Furthermore, we also find that there exist better estimators (less biased), namely the jackknifed (JK) estimator (Quenouille, 1956; Tukey, 1958) and the best-upper-bound (BUB) estimator (Paninski, 2003), and they are shown to be more consistent than MLE in our experiments (see Section 7.4.5).

In the next sections, we first discuss the entropy bias problem, then we list a few alternative entropy estimators, and finally we briefly discuss and compare alternative estimators with the ML estimator.

7.4.2 Entropy Estimation

Part I: Entropy Estimator Bias

The standard entropy is defined in terms of probabilities (see Equation 7.7), however, the real probability mass function of the class/cluster variable is unknown in the WSI evaluation case. As an estimation, the standard Maximum Likelihood (ML) estimator based on normalized

[1] This baseline assigns each instance to an individual cluster.

empirical frequency is adopted (Equation 7.9).[2]

$$\hat{H} = -\sum_{i=1}^{m} \frac{n_i}{N} \log \frac{n_i}{N} \tag{7.9}$$

where n_i is the number of instances in class/cluster i, m is the total number of classes/clusters, and N is the total number of instances (i.e., the sample size).

Now, we use the ML estimator to analyze the SemEval 2010 one-cluster-per-instance baseline (1cl1inst), which when averaged over all the 100 words significantly outperforms the best participating system on the standard test set. We calculate a general case of this baseline as in (7.10).

V-Measure Score of the 1cl1inst Baseline

Suppose the number of instances in the test set is N, then:

- The class entropy is $\hat{H}(c)$;
- The cluster entropy is,
  $\hat{H}(k) = -\sum_{i=1}^{|k|} \frac{n_i}{N} \log \frac{n_i}{N} = -\sum_{i=1}^{N} \frac{1}{N} \log \frac{1}{N} = \log N$;
- The joint entropy of the class and the cluster is,
  $\hat{H}(c,k) = -\sum_{j=1}^{|c|} \sum_{i=1}^{|k|} \frac{n_{ij}}{N} \log \frac{n_{ij}}{N} = -\sum_{i=1}^{N} \frac{1}{N} \log \frac{1}{N} = \log N$;

Therefore, the V-Measure is estimated as:

$$\hat{V}(c,k) = \frac{2\hat{H}(c)}{\log N + \hat{H}(c)} \tag{7.10}$$

As the estimated value of the class entropy $\hat{H}(c)$ on a given test set is a constant, Equation 7.10 suggests that the estimated V-Measure score of the 1cl1inst baseline is inversely proportional to the test set size (N). Therefore, the V-Measure score goes to 0 as the test size goes to infinity $N \to +\infty$:

$$\lim_{N \to +\infty} \hat{V}(c,k) = \lim_{N \to +\infty} \frac{2\hat{H}(c)}{\log N + \hat{H}(c)} = 0 \tag{7.11}$$

[2] We use ˆ to denote the estimated value.

To summarize, Equation 7.10 suggests: i) The performance of the 1cl1inst baseline is dependent on the test size; ii) The estimated V-Measure score constantly drops as the number of the instances in the test set increases. Furthermore, the performance of the 1cl1inst baseline reaches its worst point when the test set consists of infinite number of instances (see Equation 7.11). In other words, the V-Measure is positively biased; it overrates the 1cl1inst baseline on a finite test set.

We conduct further studies and find that the V-Measure problem is caused by the bias problem of the MLE. More specifically, the expected value of the entropy estimator on a finite sample set is different from the real value, and it is also different from an expected value on a larger test set generated from the same distribution. The bias problem has been noted in previous work: (i) Miller (1955) suggests that the standard ML entropy estimator is negatively biased, and he also points out that the discrepancy is proportional to the number of classes/clusters m and inversely proportional to the sample size N; (ii) Paninski (2003) further notes that the ML estimators are extremely inaccurate when class/cluster size m is comparable with sample size N.

Unfortunately, we face those exact conditions which influence the entropy estimator bias in WSI evaluation: First, the probability mass of the class/cluster distribution is unknown, which leads to the usage of empirical frequency counts in replacement of probabilities (where entropy estimator bias is introduced). Second, The number of test instances per word is very small compared with the number of classes, clusters or the combinations of the two (joint entropy). This leads to imprecise estimation of

- the class marginal entropy $\hat{H}(c)$
  e.g., 10 gold sense classes v.s. 28 test instances (for target noun *screen*, SemEval 2010);
- the cluster marginal entropy $\hat{H}(k)$
  e.g., 351 clusters by UoY v.s. 31 test instances (noun *accounting*, SemEval 2010);
- the worst of all, the class and cluster joint entropy $\hat{H}(c, k)$
  e.g., 1755 class-cluster pairs by UoY v.s. 31 test instances (*accounting*, SemEval 2010).

Since the ML estimator introduces bigger errors to the joint entropy estimator $\hat{H}(c, k)$ than to the marginal entropy estimator $\hat{H}(k)$ (i.e., the absolute value of the bias for the joint entropy estimator exceeds the marginal entropy estimator), the V-Measure is positively biased (see (Miller, 1955) for the relation between entropy bias and number of classes/samples). As the bias

would be extremely high for systems predicting many clusters, the V-Measure problem is not only for the 1cl1inst baseline but also for any system which outputs a large number of clusters.

Part II: Alternative Entropy Estimators

In this part, we first review the standard Maximum Likelihood estimator, and then introduce alternative estimators which contain bias correction terms and can better serve the WSI evaluation task.

1. Maximum Likelihood Estimator (MLE) (Strong et al., 1998) estimates the entropy of a distribution p drawn on N samples as:

$$\hat{H}_{MLE}(p_N) = -\sum_{i=1}^{m} p_{N,i} \log p_{N,i} \tag{7.12}$$

   where N is the sample size; m is the number of classes; $p_{N,i}$ is the probability of observing class i within the samples, and it is usually estimated by empirical frequency counts.

2. Miller-Madow's Estimator (MM) (Miller, 1955) introduces a bias correction factor which is based on the observation that MLE is negatively biased, and this bias increases as the number of classes m grows and decreases as the number of samples N grows. As a result, the Miller-Madow bias correction factor is defined as proportional to the number of classes and inversely proportional to the number of samples.

$$\hat{H}_{MM}(p_N) = \hat{H}_{MLE}(p_N) + \frac{\hat{m} - 1}{2N} \tag{7.13}$$

   where $\hat{m}$ is the estimated number of classes.

3. Jackknifed Estimator (JK) (Quenouille, 1956; Tukey, 1958) also notes that the discrepancy of the true entropy and the estimator varies as the sample size changes. Intuitively, the discrepancy can also be utilized to correct bias: if an estimated value based on N samples exceeds the estimated value from N-1 samples, then the estimator bias is big.

$$\hat{H}_{JK}(p_N) = N\hat{H}_{MLE}(p_N) - \frac{N-1}{N} \sum_{j=1}^{N} \hat{H}_{MLE}(p_{N,-j}) \tag{7.14}$$

   where $p_{N,-j}$ is all the samples excluding the j^{th} one.

4. "Best Upper Bounds" Estimator (BUB) (Paninski, 2003) defines the *histogram order statistics* h_j as the number of classes that appear j times in the samples (Equation 7.15), and he notes that each entropy estimator $\hat{H}$ (ML)/$\hat{H}$(MM)/$\hat{H}$(JK) is a linear function of the histogram order statistics (Equation 7.16), however, they are different in selecting the coefficient $a_{\hat{H},j,N}$. The BUB estimator sets the coefficient $a_{\hat{H}_{BUB},j,N}$ by optimizing the best upper bounds and formulating it as a regularized least-squares problem. We refer the reader to (Paninski, 2003) for details. In this thesis we focus on the application of the BUB estimator to the WSI evaluation problem.

$$h_j = \sum_{i=1}^{m} [[n_i = j]], \qquad [[a = b]] = \begin{cases} 1 & \text{if a=b} \\ 0 & \text{if a}\neq \text{ b} \end{cases} \tag{7.15}$$

where n_i is the number of times that class i appears in the sample.

$$\hat{H}_{j,N} = \sum_{j=o}^{N} a_{\hat{H},j,N} h_j \tag{7.16}$$

Part III: Discussion

In general, the MM, JK and BUB estimators are more favorable than the ML estimator as they add extra bias correction factors. Our first question is how well these different bias correction estimators perform compared with each other.

The BUB estimator is considered to be particularly effective for situations in which the sample size is comparable to the number of classes $N \sim m$, or situations in which the distribution that generates the samples is highly skewed (e.g., Zipf distribution). On the other hand, WSI evaluation entropy estimation exhibits two special properties: 1) There are many systems outputting a large number of induced clusters while the test instance size is relatively small. 2) Word sense distribution is highly skewed as the most frequent sense occurs predominantly (McCarthy, 2009). Our second question is whether BUB is particularly suitable for WSI evaluation.

In the experiment sections, we answer these two questions: First, we use uniform and Zipf distributions to randomly generate 1000 samples; Then, we use different estimators discussed in this section to run entropy estimation and compare them with the true entropy; Finally, we discuss the performance of different estimators and their potential effects on WSI evaluation.

7.4.3 Stochastic Predictions

As SemEval WSI organizers encouraged participants to output clusters with weighted coefficients for evaluation, two of the systems of SemEval 2010 actually submitted weighted results (KCDC-PC-2 and UoY). The weight information was utilized for the supervised evaluation, but it was discarded for the V-Measure evaluation. In fact, the V-Measure evaluation of SemEval 2010 only chose the highest weighted cluster as the final output and estimated the entropy based on this single cluster output.[1] In our study, however, we find weighted clusters contain rich information that is worth exploring for evaluation, therefore, we develop an approach to integrate weighted cluster output into the entropy estimators. We describe the stochastic prediction process that we adopt to estimate the entropy of weighted clusters in this section.

As in the uniform representation of different entropy estimators (Equation 7.16), given a specific estimator coefficient $a_{\hat{H},j,N}$, we need to estimate the histogram order statistics h_j (the number of classes that appear j times in the samples). Due to the linearity of the estimator, we can rewrite the expected value of the estimator $E(\hat{H})$ by the expected value of the h_j variable times the constant matrix $a_{\hat{H}}$:

$$E(\hat{H}) = \sum_{j=o}^{N} a_{\hat{H}} E(h_j) \tag{7.17}$$

where $E(h_j)$ can be further rewritten as:

$$E(h_j) = \sum_{i=1}^{m} P(n_i = j, N) \tag{7.18}$$

where $P(n_i = j, N)$ represents the probability of class i occurring j times in N samples.

The rest of the problem is to estimate $P(n_i = j, N)$, typical non-identical Bernoulli trials, also known as the Poisson Binomial distribution (Wang, 1993). The problem is to estimate the probability of having j successful trials of class i out of a total number of N trials. The direct calculation of such P is intractable but there exist efficient recursive forms. We choose the one proposed by Gail et al. (1981): The event that class i occurs j times out of N trials equals the event that class i occurs $j-1$ times out of the first $N-1$ trials and it occurs again in the N^{th} trial. Alternatively, it also equals the event that class i occurs j times out of the first $N-1$ trials and it does not occur in the N^{th} trial. The process is represented as:

$$\begin{aligned} P(n_i = j, N) = {} & P(n_i = j-1, N-1) \times p_i^{(N)} \\ & + P(n_i = j, N-1) \times p_{\bar{i}}^{(N)} \end{aligned} \tag{7.19}$$

[1] Whenever there exists a tie, the first appeared cluster was chosen.

where $p_i^{(N)}$ is the probability of observing class i in the N^{th} trial, $p_{\bar{i}}^{(N)}$ is the probability of not observing class i in the N^{th} trial, $p_{\bar{i}}^{(N)} = 1 - p_i^{(N)}$.

According to Equation 7.19, the estimation is a dynamic programming process (Bellman, 2010) in which the initial values are set to be:

- $P(n_i = 0, 0) = 1$, the event of observing class i zero times out of zero samples is always true;
- $P(n_i = -1, 0) = 0$, the event of observing class i minus one time out of zero samples is impossible.

We then utilize those initial values to calculate subsequent numbers. For instance, $P(n_i = 0, 1)$, the probability of observing class i zero times out of one sample, can also be calculated as the recursive process:

$$\begin{aligned} P(n_i = 0, 1) &= P(n_i = -1, 0) \times p_i^{(1)} + P(n_i = 0, 0) \times (1 - p_i^{(1)}) \\ &= 0 \times p_i^{(1)} + 1 \times (1 - p_i^{(1)}) \\ &= 1 - p_i^{(1)} \end{aligned} \tag{7.20}$$

The results $P(n_i = 0, 1) = 1 - p_i^{(1)}$ suggests that the probability of observing class i zero times out of one samples is the probability that the first sample is not from class i. This exactly confirms our intuition.

Similarly, the probability of observing class i once out of one samples, $P(n_i = 1, 1)$, is the same as the probability of the first sample belonging to class i (Equation 7.21).

$$\begin{aligned} P(n_i = 1, 1) &= P(n_i = 0, 0) \times p_i^{(1)} + P(n_i = 1, 0) \times (1 - p_i^{(1)}) \\ &= 1 \times p_i^{(1)} + 0 \times (1 - p_i^{(1)}) \\ &= p_i^{(1)} \end{aligned} \tag{7.21}$$

Now we instantiate Equation 7.19 with the concepts in WSI evaluation: N is the number of instances in the evaluation test set; $P(n_i = j, N)$ is the probability that j out of N instances belong to cluster i; $p_i^{(N)}$ is the normalized weight of cluster i assigned to the N^{th} instance. After the probability estimation of the histogram order statistics h_j, we can use Equation 7.17 combined with $a_{\hat{H}}$ from different entropy estimators (ML, MM, JK, BUB) to estimate the entropy and apply V-Measure to rank the systems.

7.4.4 Experiment 1: Simulations

We run simulation experiments on samples that are generated from uniform (Johnson, 1994) and Zipf (Zipf, 1935) distributions of which we know the true entropies. We select uniform distribution as it is the maximum entropy distribution among all discrete distributions supported on this set and has been frequently adopted to various problems (Park and Bera, 2009). We select Zipf distribution as it is very similar to the skewed long-tailed distribution of WSI (McCarthy, 2009), thus, the performance of entropy estimators on Zipf distribution is indicative of how well they might perform on WSI evaluation.

The probability mass function of uniform distribution is $p(m_i) = \frac{1}{m}$, i.e., each class has an equal probability of generating a sample. The true entropy of uniform distribution can be easily calculated as $\ln(m)$.

Zipf's distribution suggests that lower ranked classes (bigger k) occur very infrequently , and the probability mass function is defined as:

$$f(k; s, m) = \frac{1}{k^s H_{m,s}}, \qquad H_{m,s} = \sum_{i=1}^{m} 1/i^s \tag{7.22}$$

where $f(k; s, m)$ is the number of classes of rank k; m is the number of classes; s is the parameter characterizing the skew degree of the distribution.

The entropy of Zipf distribution is dependent on two parameters, the number of classes m and the skew degree s. It can be calculated as:

$$E(Zipf(m, s)) = \ln H_{m,s} + \frac{s}{H_{m,s}} \sum_{i=1}^{m} \frac{\ln(i)}{i^s} \tag{7.23}$$

In the simulation experiments, we first use the probability mass function of the two distributions to randomly generate samples; Then, we use the samples to estimate the entropy based on the 4 different entropy estimators discussed in the last section (ML, MM, JK, BUB); Finally, we compare the estimated entropy with the true entropy (H). We set the number of classes m as 10 and vary the sample size N. We also average over 1000 rounds to make the simulation more reliable when generating samples.

The uniform simulation experimental results are plotted in Figure 7.1. As we can see, the standard Maximum Likelihood entropy estimator ML is negatively biased, and this bias tends to be huge when the sample size N is small ($N < 3$). The bias reduces as more and more samples are used. We can also see that all bias correction estimators (MM, JK, BUB) perform better

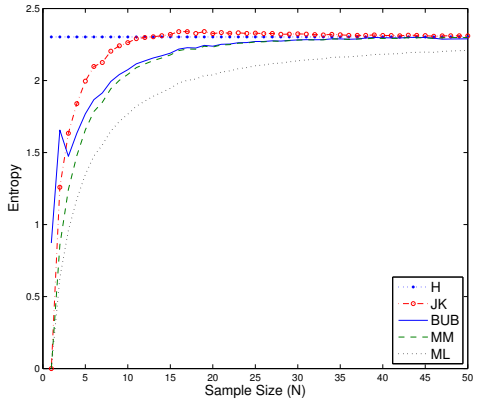


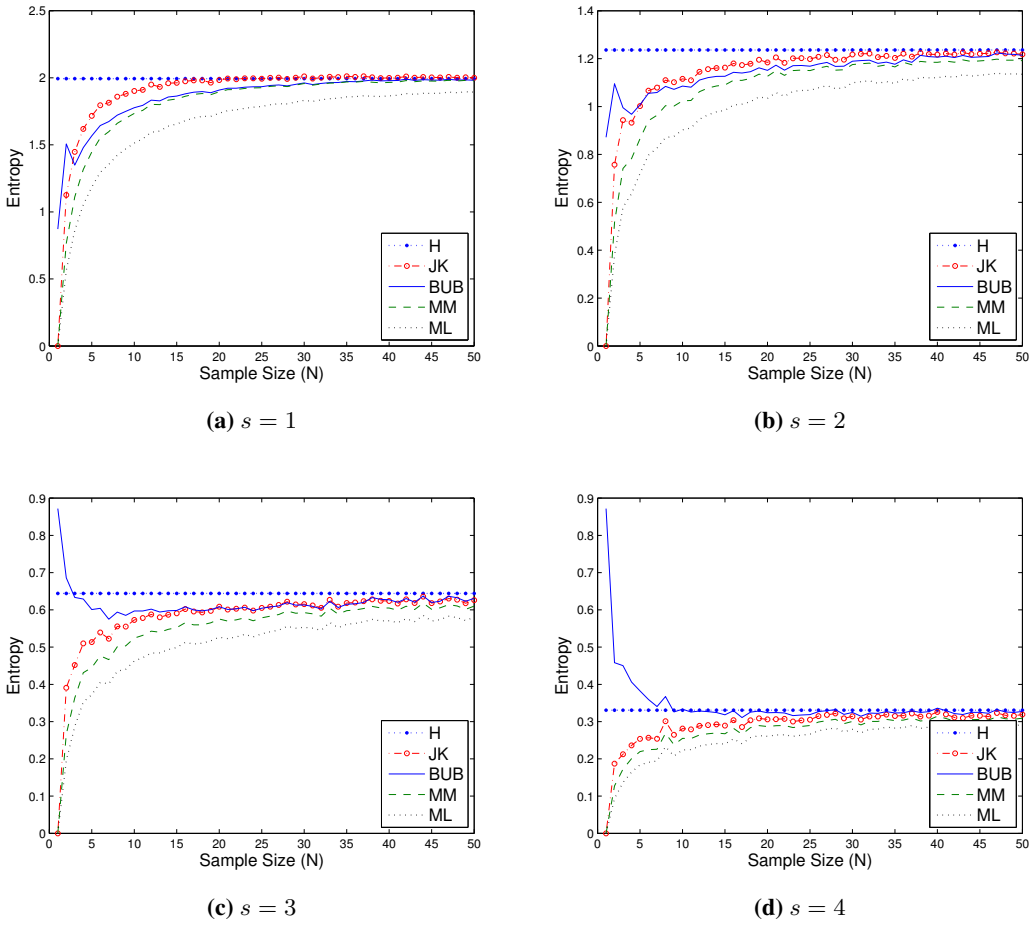
Figure 7.1: The estimated (ML, MM, JK, BUB) and true entropy for discrete uniform distribution, the number of classes is set to be $m = 10$, natural logarithm $\ln$ is adopted.

than the ML estimator. However, the MM estimator has a mild correction factor which does not sufficiently correct the bias, especially when compared to the other two estimators (JK and BUB). We also find that BUB has a more effective bias correction factor when the sample size is very small, and it outperforms all the other estimators at the initial stage ($N < 3$). However, this advantage is eliminated when the sample size grows as the JK estimator gains the best performance ($N > 4$).

The Zipf distribution simulation experiments are plotted in Figure 7.2. When the skew factor of the Zipf distribution is small ($s = 1, 2$), we observe similar patterns as the uniform simulation: 1) JK and BUB are the two favorable estimators; 2) BUB performs the best when the sample size is very small, and JK outperforms others when the sample size grows bigger. When the distribution is more skewed ($s = 3, 4$), we find that BUB stably outperforms JK for cases where $N > 3$, however, we notice some inconsistent behavior of BUB when the sample size is very small ($N < 3$): BUB over-corrects the bias and leads to positive bias.[1]

In all, the standard ML estimator is unfavorable compared with estimators with bias correction factors. We show that the MM estimator has a very mild bias correction factor, while JK and BUB estimators are likely to more sufficiently correct the bias. We also show that BUB tends to perform better on a very skewed distribution on a relatively small number of samples, but it has some inconsistent behavior on skewed distributions when sample size N is less than

[1]This inconsistency leads to some unexpected behavior of the BUB estimator on WSI evaluation experiments. We come back to this problem in Section 7.4.5.



(a) $s = 1$ **(b)** $s = 2$

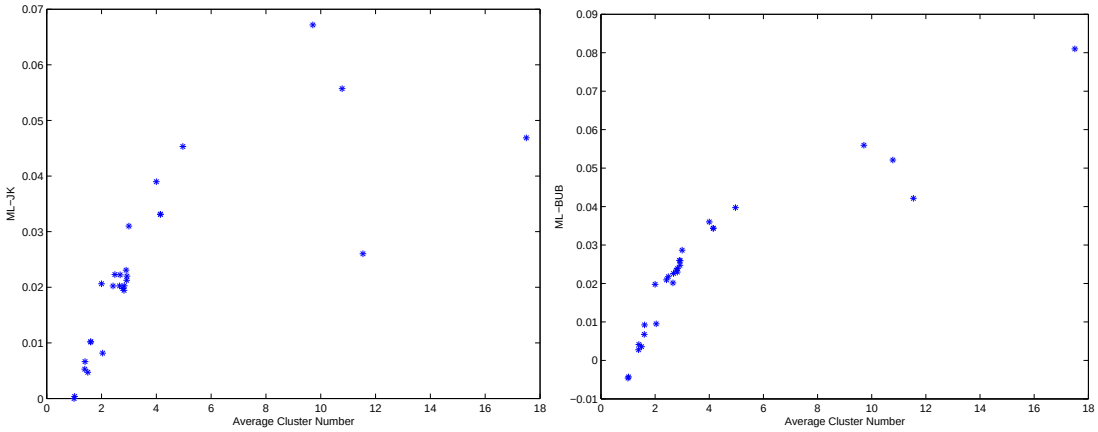
(c) $s = 3$ **(d)** $s = 4$

Figure 7.2: The estimated (different entropy estimators) and true entropy of Zipf's law, the number of classes is set to be $m = 10$, natural logarithm $\ln$ is adopted.

three. Finally, we show that JK consistently achieves good performance on various different distributions, clearly more favorable than ML/MM and more consistent than BUB (small sample size for skewed distributions).

7.4.5 Experiment 2: Effects on WSI Evaluation

In this section, we discuss the effects on the SemEval 2010 WSI shared task by applying different entropy estimators. The data set is described in Section 2.4. Overall, 26 systems participate in the task and there are also three baselines: (1) MFS (most frequent sense) assigns every instance



(a) C# v.s. ($\hat{V}_{ML}$ - $\hat{V}_{JK}$)

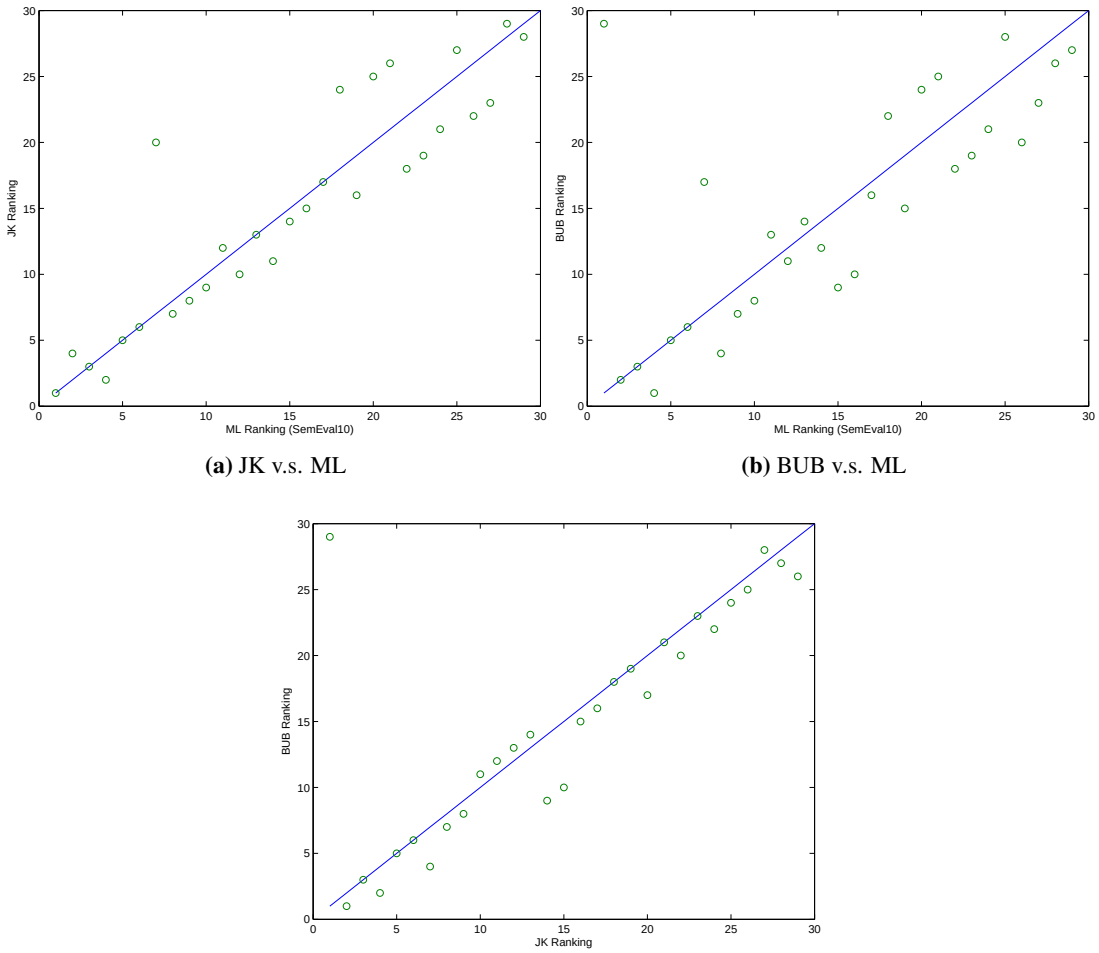
(b) C# v.s. ($\hat{V}_{ML}$ - $\hat{V}_{BUB}$)

Figure 7.3: Discrepancy in entropy estimators (V-Measure) as function of the predicted number of clusters. The dots in the figures represent different systems of SemEval 2010.

of a lemma to the same cluster; 2) 1cl1inst (one cluster per instance) assigns each instance to an individual cluster; 3) Random (choose cluster randomly) assigns each instance to a random cluster. In the shared task, systems were ranked according to different evaluation measures such as *F-Score*, *supervised*, and *V-Measure*, however, we only focus on the study of the information theoretic approach V-Measure, a measure based on entropy estimation.

As we have already discussed in Section 7.4.2 and 7.4.4, the number of clusters influences the bias of the entropy estimation which consequently also affects the bias of the V-Measure. Our first experiment is to check the discrepancy of the V-Measure of different estimators versus the average number of clusters, ranging from 1.02 (Duluth-WSI-SVD) to 17.5 (KSU). The results are shown in Figure 7.3. We find that the discrepancy between the improved estimators (JK or BUB) and the standard ML estimator increases as the number of clusters grows. We also find that the V-Measure discrepancy between ML and BUB is as high as 0.08 when the average number of clusters is around 18 (KSU) (Figure 7.3b). These experimental results show that the V-Measure based on ML estimator is unreliably positively biased; and furthermore, the bias is correlated to the number of clusters: the more clusters a system outputs, the higher bias the V-Measure has.

The second experiment is to determine how the ranking is affected by using different entropy estimators. We plotted different ranking comparisons as in Figure 7.4 (JK v.s. ML, BUB v.s.



(a) JK v.s. ML

(b) BUB v.s. ML

(c) BUB v.s. JK

Figure 7.4: Discrepancy in rankings by different entropy estimators. Circles in the figure represent different systems, x axes is ranking by one estimator, y axes is the ranking by another estimator.

ML, and BUB v.s. JK). If two entropy estimators agree with each other then circles should all fall on the diagonal line $x = y$, i.e., the ranking of one estimator is exact the same as that of the other estimator. However, our experiments show that different estimators do not agree with each other perfectly. We find that the JK and BUB estimators rank systems differently from the ML estimator (Figure 7.4a and 7.4b), while, in contrast, the BUB estimator and the JK estimator achieve better agreement as most of the circles are distributed around the diagonal line (Figure 7.4c). The fact that the two bias correction estimators (JK, BUB) reach better agreement compared with JK and ML, or BUB and ML, further supports one of the main arguments in this

chapter that the standard ML estimator needs to be replaced by more precise estimators for WSI evaluation.

System	C#	ML		MM		JK		BUB	
		sc.	r#	sc.	r#	sc.	r#	sc.	r#
1cl1inst	89.1	31.6	1	29.5	1	27.4	1	-3.6	29
Hermit	10.8	16.2	2	13.1	4	10.7	4	11.0	2
UoY	11.5	15.7	4	14.3	2	13.1	2	11.4	1
KSU KDD	17.5	15.7	3	13.2	3	11.0	3	7.6	3
Duluth-WSI	4.1	9.0	5	6.9	5	5.7	5	5.6	5
Duluth-WSI-SVD	4.1	9.0	6	6.9	6	5.7	6	5.6	6
Duluth-R-110	9.7	8.6	7	4.7	16	1.9	20	3	17
Duluth-WSI-Co	2.5	7.9	8	6.4	7	5.7	7	5.7	4
KCDC-PCGD	2.9	7.8	9	6.3	8	5.5	8	5.2	7
KCDC-PC	2.9	7.5	10	6.2	9	5.4	9	5.0	8
KCDC-PC-2	2.9	7.1	11	5.7	12	4.9	12	4.5	13
Duluth-Mix-Narrow-Gap	2.4	6.9	15	5.5	14	4.8	14	4.8	9
KCDC-GD-2	2.8	6.9	14	5.7	11	4.9	11	4.6	12
KCDC-GD	2.8	6.9	12	5.8	10	5.0	10	4.6	11
Duluth-Mix-Narrow-PK2	2.7	6.8	16	5.4	15	4.6	15	4.6	10
Duluth-Mix-PK2	2.7	5.6	17	4.3	17	3.5	17	3.5	16
Duluth-R-15	5.0	5.3	18	2.4	20	0.7	24	1.3	22
Duluth-WSI-Co-Gap	1.6	4.8	19	4.1	18	3.8	16	4.1	15
Random	4.0	4.4	20	1.9	22	0.5	25	0.8	24
Duluth-R-13	3.0	3.6	21	1.5	25	0.5	26	0.7	25
Duluth-WSI-Gap	1.4	3.1	22	2.6	19	2.5	18	2.7	18
Duluth-Mix-Gap	1.6	3.0	23	2.3	21	1.9	19	2.0	19
Duluth-Mix-Uni-PK2	2.0	2.4	24	1.8	23	1.5	21	1.4	21
Duluth-R-12	2.0	2.3	25	0.8	27	0.2	27	0.3	28
KCDC-PT	1.5	1.9	26	1.6	24	1.4	22	1.5	20
Duluth-Mix-Uni-Gap	1.4	1.4	27	1.0	26	0.8	23	1.0	23
KCDC-GDC	2.8	6.9	13	5.7	13	4.8	13	4.5	14
MFS	1.0	0	29	0.0	29	0.0	28	0.5	27
Duluth-WSI-SVD-Gap	1.0	0.0	28	0.0	28	0.0	29	0.5	26
KCDC-PC-2*	2.9	5.7	-	7.2	-	2.3	-	2.2	-
UoY*	11.5	25.1	-	22.8	-	17.8	-	5.0	-

Table 7.1: The percentage V-measure computed with different estimators and the corresponding rank. C# is the average number of clusters. ML is the maximum likelihood estimator. MM is the Miller-Madow estimator. JK is the jackknifed estimator. BUB is the best upper bound estimator. "sc." is the score. "r#" is the rank. KCDC-PC-2* and UoY* are from stochastic prediction.

In the last experiment, we give the details of the scores and rankings produced by different estimators on the SemEval 2010 WSI task. The systems are presented in the order in which they were given in the SemEval 2010 result table (Manandhar et al., 2010, see p. 66, Table 4).[1] We show the results in Table 7.1, which contains the average number of clusters (C#), the V-Measure score computed by different estimators (sc.) and the ranking produced by these estimators (r#).

Table 7.1 shows that the ML estimator ranks the 1cl1inst baseline the best (a V-Measure score of 31.6, significantly higher than any other system). The bias correction estimators (MM, JK and BUB) score this baseline lower to different extent: While the MM and JK estimators maintain a high score (MM 29.5, JK 27.4), the BUB estimator, which is specially designed for cases in which the number of classes is comparable to sample size $m \sim N$, reduces this bias significantly and ranks the 1cl1inst baseline last. One surprising result that we noticed is that the V-Measure score estimated by BUB on the 1cl1inst baseline is negative. We find that it is caused by the fact that the BUB estimator over-corrects the negative bias and leads to positive bias when the sample size is considerably smaller than the number of clusters for very skewed distributions (See Figure 7.2d). In the 1cl1inst baseline case, the joint entropy of classes and clusters $H(c, k)$ faces this situation. As a result of the excessive positive bias estimated by the BUB estimator on $H(c, k)$, the V-Measure score is negatively biased to a minus value. However, it is important to notice that for the vast majority of the systems there is agreement between the JK and BUB estimators, whereas the ML estimator significantly overestimates the V-Measure. This observation coupled with the observed behavior of the JK and BUB estimators in the simulations suggests that JK and BUB are considerably more reliable than the state-of-the-art ML estimator.

The bottom two systems of Table 7.1 (KCDC-PC-2* and UoY*) are computed from the stochastic prediction as described in Section 7.4.3 (used for weighted cluster output)[1]. Note that the comparison between the stochastic version and the unweighted version is unnecessary (KCDC-PC-2 v.s. KCDC-PC-2*, UoY v.s. UoY*), since the unweighted version is quite

[1]The ranking produced by the ML estimator should mirror that of the official results. In some cases it does not, e.g. system UoY was placed before KSU in the official results, while the ML estimator would predict the reverse order. As the difference in V-measure is small, we attribute this discrepancy to rounding errors. Our ranking were computed before rounding, and there were no ties. The system KCDC-GDC seems to be misplaced in the official results list; according to V-measure it should be ranked higher.

[1]All the other systems did not submit weighted cluster output. We do not include the two stochastic predictions in the ranking.

different from the weighted one as it truncates the most weighted cluster even if there exists a long tie (e.g., 10 clusters with the same weights). Although we did not include the weighted version in the ranking of SemEval 2010 task as most systems did not submit weighted cluster output, we believe the stochastic predictions proposed in this section would be very useful for future WSI evaluation as weighted cluster output is a trend encouraged by the SemEval WSI task organizers (Agirre and Soroa, 2007).

We also find that none of the estimators can reliably estimate the entropy when the sample size compared with the number of classes/clusters (N/m) is very small (Figure 7.2). Specifically, the bias is very big when the instance size is smaller than the number of classes $N/m < 1$. This bias is consistently negative for ML/MM/JK, and it is negative for BUB on uniform distribution and flat Zipf distributions ($s = 1, 2$) but positive on more skewed Zipf distributions ($s = 3, 4$). When the sample size is relatively big compared to the number of classes ($N/m > 1$), all estimators, while being consistently negatively biased, can fairly reliably estimate the entropy, i.e., the estimated value is very close to the true entropy. Therefore, we propose that future WSI evaluation should constrain the size of test instances to be more than the number of classes $N/m > 1$. This is to say that the more samples are there in the test set the more clusters can be output by those systems. Quantitatively, we apply the constraint on the marginal and joint entropy estimation as follows:

1. Class marginal entropy estimation $\hat{H}(c)$: $\frac{N}{c} > 1 \Rightarrow N > c$.
   i.e., test set size should be more than the number of gold classes.

2. Cluster marginal entropy estimation $\hat{H}(k)$: $\frac{N}{k} > 1 \Rightarrow k < N$.
   i.e., the maximum number of clusters outputted by system should be less than test set size.

3. Class and cluster joint entropy estimation $\hat{H}(c, k)$: $\frac{N}{(c,k)} = \frac{N}{c \times k} > 1 \Rightarrow k < \frac{N}{c}$.
   i.e., the maximum number of clusters should be less than test size divided by the number of gold classes.

To summarize, in this section we argue that future WSI tasks should use more precise entropy estimators (JK and BUB), and furthermore, they should fulfill two principles in order to adopt the information theoretic based V-Measure: First, they should supply a test set of which the size is bigger than the number of gold classes $N > c$; Second, the maximum number of clusters output by a system should be constrained to less than the test set size divided by the number of gold classes $k < N/c$.

7.4.6 Conclusion

In this section, we discuss the entropy bias problem of the information theoretic V-Measure evaluation approach. We find that the state-of-the-art ML estimator is biased and conduct studies on alternative entropy estimators with bias correction factors. We find two alternative estimators (JK and BUB) perform very well in simulation experiments in which we compare the estimated entropy against the true entropy of the uniform and Zipf distributions. We also find that there is a general agreement between the JK and BUB estimator, while the discrepancy between JK and ML, or BUB and ML, is high. We argue that the more precise estimators should replace the standard ML estimator for future WSI evaluation tasks.

Furthermore, we notice that all the estimators, including the bias correction ones, are not able to effectively deal with situations in which the number of instances is smaller than the number of classes/clusters. We also find that the joint entropy estimation of class and cluster pair (c, k) is the main bottleneck of the problem. We propose future WSI tasks should constrain the number of clusters and test set size so that the total number of instances is always higher than the number of classes/clusters/(class, cluster)pairs. More specifically, the test size should be higher than the number of gold classes, and the maximum number of clusters allowed by systems should be less than the instance size divided by the number of gold classes.

In addition, we propose a stochastic prediction measure of how to incorporate weighted cluster output into the evaluation model. It is the first time, as far as we know, that weighted class problem is addressed in the V-Measure evaluation. We believe this work would contribute to the study of WSI because weighted cluster output is a natural approach to pursue in future study.

7.5 Summary

In summary, this chapter studies a lexical ambiguity task in which the sense inventory is underspecified (WSI). We find that state-of-the-art WSI research is hindered by the deficiencies of the commonly used evaluation approaches. The work described in this chapter proposes solutions to fix those evaluation problems and set a fairer platform for comparisons among different approaches in future research.

We show that the state-of-the-art supervised WSI evaluation has three problems: (1) The mapping matrix backs off the system result to the distribution in the training set, thereby

compressing the results of all the systems into a narrow band that converges around the most-frequent-sense baseline; (2) It encourages to over-generate a large number of clusters, as it utilizes the information in the mapping matrix (from the training set) to tune the best linear combination of clusters to map senses; (3) The ranking produced by such an evaluation measure outputs inconsistent rankings as the training-test split varies.

Various unsupervised evaluation approaches have also been proposed but most of them fail to distinguish the Semeval 2010 systems from the baselines. The Fscore, Entropy and Purity approaches implicitly encode mapping schemes between senses and clusters, allowing either one-to-many or many-to-one mapping. These mapping based approaches either encourage a very coarse-grained (e.g., most-frequent-sense) or a very fine-grained (e.g., one-cluster-per-instance) output.

The V-Measure avoids the mapping process, which is either explicitly or implicitly utilized in the supervised measure and most of the unsupervised measures. It is essentially normalized mutual information that measures the statistical dependency between the gold class variable and the system cluster variable. In our study, we find that the entropy estimator is negatively biased, which consequently leads to the positive bias of the V-Measure. Systems that output a large number of clusters benefit from the entropy bias. As a result, the one-cluster-per-instance baseline was ranked first in the SemEval 2010 WSI task. Further study suggests that there exist more reliable entropy estimators with bias correction factors, which are shown to be more effective than the standard ML estimator in our experiments. We argue that the more precise estimators (JK and BUB) should replace ML for future WSI evaluation. We also show that all the investigated entropy estimators are unreliable when the sample size is smaller than the number of clusters. Therefore, we further suggest that WSI evaluation should constrain the number of clusters allowed by system to be less than the test size divided by the number of gold-standard classes.

8

Conclusion

8.1 Summary

In this thesis, we have proposed computational approaches to model various lexical ambiguity phenomena in natural language, such as idiomatic expression detection (e.g., *spill the beans*), novel figurative expression detection (e.g., *take the sock out of your mouth*), word sense disambiguation (e.g., *bank* as "a financial institute"), and word sense induction (e.g., *shake/cluster*$_n$).

From the language phenomena point of view, we have presented our work on problems of which the sense granularity change from more clearly distinguishable categorizations (e.g., in the idiom detection task, *spill the beans* 'literal' v.s. *spill the beans* 'idiomatic'), to the more fuzzy ones (e.g., in the fine-grained word sense disambiguation task, WordNet sense 'bank%1:14:00::' as "depository financial institution", '*bank%1:14:00::*' as "a financial institute", 'bank%1:17:01::' as "a sloping land", 'bank%1:04:00::' as "a flight maneuver", 'bank%1:06:00::' as "bank building", etc.), and to the underspecified ones (e.g., in the word sense induction task, the sense inventory is undefined and the number of induced clusters is underspecified, *shake/cluster*$_1$, *shake/cluster*$_2$, . . ., *shake/cluster*$_n$).

From a statistical modeling point of view, we have developed different models aiming at finding the most efficient solution to a problem with the lowest cost. These advanced statistical models reduce the annotation effort and improve performance. This can be demonstrated in particular by the idiom detection task: i) We started with a supervised model which outperforms state-of-the-art systems (Chapter 3). ii) In the next chapter, we improved the model to reduce annotation work by providing a completely unsupervised model that adopts a bootstrapping

strategy, while maintaining a high performance. iii) Chapter 6 further improved the performance by taking a more sophisticated statistical modeling approach which incorporates easily-obtained human knowledge into the probability models. In addition, we experimented with modeling different lexical phenomena within a common statistical framework such as the topic model used for both idiom detection and WSD (Chapter 6). We find that our proposed models outperform state-of-the-art systems.

The individual chapters are summarized as follows:

Chapter 3 proposes a supervised model to detect idiomatic expressions. We experiment with various features, such as global context, local context, lexical cohesion based features, syntactic features, named entity features and features exploit indicative lexical terms. We also experiment with idiom specific models, generic models, and models that generalize over unseen idioms. We find that the statistical type features, bag-of-words contexts and lexical cohesion based features work the best for distinguishing idiomatic usages from literal readings. Certain linguistic features further boost the performance. However, linguistic features are very sparse and are not very useful when the distribution of the true label is heavily imbalanced (e.g., idiomatic usage occurs predominately). We also found that lexical cohesion features have the best generalization ability, and they can be used to discover new idiomatic expressions.

Chapter 4 is built upon the work of the previous chapter. It aims at solving the same problem using an unsupervised approach to reduce human annotation. We present a bootstrapping strategy which is built on an unsupervised classifier and a supervised classifier. We show that such a bootstrapping strategy leads to a very good performance even compared with the fully supervised one. We also show that the performance can be further improved by iteratively increasing the minority class instances (literal cases) in the bootstrapping loop. We propose a new method for extracting minority class instances automatically by retrieving the non-canonical variant of the target idioms from raw corpus data.

Chapter 5 extends the lexical phenomenon from idiomatic expressions to general figurative expression (e.g., *rock the boat* v.s. *take the sock out of your mouth*). We propose a Gaussian Mixture Model and find that the GMM, estimated using EM, can be utilized to effectively discover new figurative expressions. We also find that the GMM performance can be further improved by using a small annotated data set to estimate the Gaussian components, means and covariances. Interestingly, we find that the estimation of the GMM is not dependent on the precise form of the figurative expression, as shared lexical cohesion features can effectively discover new figurative expressions.

Chapter 6 deals with three types of lexical ambiguity phenomena: fine-grained word sense disambiguation, coarse-grained word sense disambiguation, and idiomatic expression detection. We model the three lexical phenomena in a uniform probability framework, which is based on topic models. The basic idea of the framework is to represent senses by sense paraphrases, and maximize the conditional probability of a sense paraphrase given an instance context. We propose three instantiations of the model, and find that they beat the state-of-the-art systems in all three disambiguation tasks.

Chapter 7 further extends the lexical sense disambiguation problem to the lexical sense induction problem, in which the sense category is underspecified. Although such an idea points out an interesting direction for future research, as sense category definition is such a difficult task and often results in dispute among lexicographers, we found out that there are problems with the state-of-the-art evaluation approaches. The two major approaches, supervised evaluation and V-Measure evaluation, are both in favor of very fine-grained system output. Both of them rank the one-cluster-per-instance baseline above all other systems. Further study suggests that supervised evaluation is unreliable, as it uses the training set to tune the best linear combination of clusters to map senses. For the V-Measure, we find the entropy estimator is biased. We argue that more sophisticated entropy estimators (JK and BUB) should replace the standard ML estimator for WSI evaluation in the future. We also argue that the WSI task should constrain the maximum number of clusters allowed by system outputs, so that the entropy estimation can fall on the range (sample size N versus class number m) that estimators can reliably perform.

8.2 Outlooks

Prior Knowledge & Statistical Modeling In Chapter 6, we have shown how to incorporate human knowledge as probability priors into the statistical probabilistic model to boost the performance. We think future research along this line is promising. We may utilize general knowledge or domain specific knowledge to constrain the probabilistic framework, thereby guiding the inference procedure to optimal output.

Feeding Distribution to Word Sense The state-of-the-art WSI research focuses on evaluating the statistical dependency of system output clusters and gold sense classes (V-Measure). It supplies entropy estimators with data samples (test set) to obtain the estimated marginal and joint entropies. Alternatively, we could also use distributions to model those data samples and

then directly calculate the mutual information of the distributions. A similar idea is used in speech recognition (Killer et al., 2003). This approach can avoid the entropy bias problems seen with estimators. However, what type of distributions to choose to model the marginal class/cluster distribution, and the joint class and cluster pair distribution remain open problems to be solved. Furthermore, we would also need to develop techniques to estimate the parameters of the assumed class and cluster distributions. We leave those questions open to future research.

More Factors Influence Lexical Semantics While most of the current research has focused on a more closed form of semantics (pre-defined, assumed to be static), there exists a dynamic aspect of semantics that changes over time. For instance, the Chinese word 小姐 *(Miss)* means *noble young woman* in ancient Chinese, while, in contrast, it often means *prostitute* nowadays. A even more difficult problem occurs in cross-linguistic environments when there are different semantic associations with the same lexical entry. Natural language applications such as machine translation may face difficulties posed by such aspects. More challenges include individualized lexical choices (e.g., forum/review corpora processing) and different granularity mapping between concepts and vocabulary in different languages (e.g., how to translate if one lexical entry in the source language does not exist in the target language).

Figurative Expression across Time Language changes across time. Metaphorically used expressions can become more popular across time, and consequently, the metaphorical sense of such expressions may be lexicalized (included in standard dictionaries). There often exist fuzzy boundary cases of whether the entry has been lexicalized, i.e., dispute on whether certain expressions are used literally or nonliterally often exists. Although this phenomenon has been noted in linguistic studies, there is a general lack of computational studies on this. We think using computational approaches to model nonliteral language (metaphor, metonymy, irony, etc.) across the language evolution process is an interesting topic to explore in the future.

Appendix A

Sense Paraphrase Examples

A.1 Word Sense Paraphrases (e.g., *bank*)

WN SenseKey	Sense Paraphrase
bank%1:04:00::	bank. a flight maneuver; aircraft tips laterally about its longitudinal axis (especially in turning). "the plane went into a steep bank". vertical bank. a bank so steep that the plane's lateral axis approaches the vertical.
bank%1:06:00::	bank, bank building. a building in which the business of banking transacted. "the bank is on the corner of Nassau and Witherspoon". vault, bank vault. a strongroom or compartment (often made of steel) for safekeeping of valuables.
bank%1:06:01::	savings bank, coin bank, money box, bank. a container (usually with a slot in the top) for keeping money at home. "the coin bank was empty". piggy bank, penny bank. a child's coin bank (often shaped like a pig).

bank%1:14:00:: depository financial institution, bank, banking concern, banking company. a financial institution that accepts deposits and channels the money into lending activities. “he cashed a check at the bank”. “that bank holds the mortgage on my home”. credit union, a cooperative depository financial institution whose members can obtain loans from their combined savings. Federal Reserve Bank, reserve bank, one of 12 regional banks that monitor and act as depositories for banks in their region. agent bank, a bank that acts as an agent for a foreign bank. commercial bank, full service bank. a financial institution that accepts demand deposits and makes loans and provides other services for the public. state bank, a bank chartered by a state rather than by the federal government. lead bank, agent bank, a bank named by a lending syndicate of several banks to protect their interests. member bank, a bank that is a member of the Federal Reserve System. merchant bank, acquirer. a credit card processing bank; merchants receive credit for credit card receipts less a processing fee. acquirer, a corporation gaining financial control over another corporation or financial institution through a payment in cash or an exchange of stock. thrift institution. a depository financial institution intended to encourage personal savings and home buying. Home Loan Bank. one of 11 regional banks that monitor and make short-term credit advances to thrift institutions in their region. banking industry, banking system. banks collectively.

bank%1:14:01:: bank. an arrangement of similar objects in a row or in tiers. “he operated a bank of switches”.

bank%1:17:00:: bank. a long ridge or pile. “a huge bank of earth”. bluff. a high steep bank (usually formed by river erosion). sandbank. a submerged bank of sand near a shore or in a river; can be exposed at low tide.

bank%1:17:01:: bank. sloping land (especially the slope beside a body of water). “they pulled the canoe up on the bank”. “he sat on the bank of the river and watched the currents”. riverbank, riverside. the bank of a river. waterside. land bordering a body of water.

bank%1:17:02:: bank, cant, camber. a slope in the turn of a road or track; the outside is higher than the inside in order to reduce the effects of centrifugal force.

bank%1:21:00:: bank. a supply or stock held in reserve for future use (especially in emergencies). blood bank. a place for storing whole blood or blood plasma. “the Red Cross created a blood bank for emergencies”. eye bank. a place for storing and preserving corneas that are obtained from human corpses immediately after death; used for corneal transplantation to patients with corneal defects. food bank. a place where food is contributed and made available to those in need. “they set up a food bank for the flood victims”. soil bank. land retired from crop cultivation and planted with soil-building crops; government subsidies are paid to farmers for their retired land.

bank%1:21:01:: bank. the funds held by a gambling house or the dealer in some gambling games. “he tried to break the bank at Monte Carlo”.

bank%2:31:02:: trust, swear, rely, bank. have confidence or faith in. “We can trust in God”. “Rely on your friends”. “bank on your good education”. “I swear by my grandmother’s recipes”. credit. have trust in; trust in the truth or veracity of. lean. rely on for support. “We can lean on this man”. count, bet, depend, look, calculate, reckon. have faith or confidence in. “you can count on me to help you any time”. “Look to your friends for support”. “You can bet on that!”. “Depend on your family in times of crisis”.

bank%2:35:00:: bank. enclose with a bank. “bank roads”.

bank%2:35:01:: bank. cover with ashes so to control the rate of burning. “bank a fire”.

bank%2:38:00:: bank. tip laterally. “the pilot had to bank the aircraft”.

bank%2:40:00:: deposit, bank. put into a bank account. “She deposits her paycheck every month”. redeposit. deposit anew.

bank%2:40:01:: bank. be in the banking business. bank. act as the banker in a game or in gambling.

bank%2:40:02::	bank. do business with a bank or keep an account at a bank. "Where do you bank in this town?".
bank%2:40:03::	bank. act as the banker in a game or in gambling. be in the banking business.

Table A.1: An example of sense paraphrase for the word "bank". Texts are from the "word forms", "glossses" and "example sentence" fields from the *sense synset* and its *reference synsets* in WordNet 2.1.

A.2 Idiom Sense Paraphrases

Idiom	Type	Paraphrase
bite off more than one can chew	l	bite, more, chew
	n	to try to do too much; to take on or attempt more than one is capable of doing.
back the wrong horse	l	back, wrong, horse
	n	give your support to the losing side in something.
blow ones own trumpet	l	blow, trumpet
	n	boast about talents and achievements.
bite ones tongue	l	bite, tongue
	n	refrain from speaking because it is socially or otherwise better not to.
bounce off the wall	l	bounce, wall
	n	high strung, energetic, over excited.
break the ice	l	break, ice
	n	ease tensions, get people talking, facilitate communication. You get over any initial embarrassment or shyness when you meet someone for the first time and start conversing.
drop the ball	l	drop, ball

	n	not doing the job or taking the responsibilities seriously enough and let something go wrong.
get ones feet wet	l	get, feet, wet
	n	first experience, dabble, dabbling. To take a risk and try something new.
pass the buck	l	buck
	n	avoid taking responsibility by saying that someone else is responsible.
play with fire	l	play, fire
	n	risky behaviour, risky behavior, take risks, act dangerously. To put oneself in a precarious situation with a high risk of getting harmed, particularly emotionally or financially.
pull the trigger	l	pull, trigger
	n	to commit to a course of action.
rock the boat	l	rock, boat
	n	upset conventions, break norms, cause trouble, disturb balance, destabilise a situation by making trouble.
set in stone	l	set, stone
	n	it cannot be changed or altered.
spill the beans	l	spill, beans
	n	reveal a secret or confess to something.
sweep under the carpet	l	sweep, carpet
	n	to hide or ignore something.
swim against the tide	l	swim, tide
	n	to do something that is in opposition to the general movement of things.
tear ones hair out	l	tear, hair
	n	to be greatly upset or distressed.

Table A.2: Idiom Sense Paraphrases

A. SENSE PARAPHRASE EXAMPLES

References

Steven Abney. Understanding the yarowsky algorithm. *Computational Linguistics*, 30:365–395, 2004.

Eneko Agirre and Aitor Soroa. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluation*, pages 7–12, 2007.

Eneko Agirre, David Martínez, Oier López de Lacalle, and Aitor Soroa. Evaluating and optimizing the parameters of an unsupervised graph-based WSD algorithm. In *Workshop on TextGraphs at HLT-NAACL 2006*, pages 89–96, 2006.

Henry Anaya-Sánchez, Aurora Pons-Porrata, and Rafael Berlanga-Llavori. TKB-UO: using sense clustering for WSD. In *SemEval '07: Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 322–325, 2007.

Javier Artiles, Enrique Amigó, and Julio Gonzalo. The role of named entities in web people search. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 534–542, 2009.

Timothy Baldwin, Colin Bannard, Takaaki Tanaka, and Dominic Widdows. An empirical model of multiword expression decomposability. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 89–96, 2003.

Timothy Baldwin, Emily M. Bender, Dan Flickinger, Ara Kim, and Stephen Oepen. Road-testing the english resource grammar over the british national corpus. In *Proc. LREC-04*, pages 2047–2050, 2004.

Colin Bannard. A measure of syntactic flexibility for automatically identifying multiword expressions in corpora. In *Proceedings of the ACL-07 Workshop on A Broader Perspective on Multiword Expressions*, pages 1–8, 2007.

Kedar Bellare, Partha Pratim Talukdar, Giridhar Kumaran, O Pereira, Mark Liberman, Andrew Mccallum, and Mark Dredze. Lightlysupervised attribute extraction for web search. In *Proceedings of Machine Learning for Web Search Workshop, NIPS 2007*, 2007.

Richard E. Bellman. *Dynamic Programming*. Princeton University Press, New Jersey, USA, 2010.

Steven Bethard, Vicky Tzuyin Lai, and James H. Martin. Topic model analysis of metaphor frequency for psycholinguistic stimuli. In *CALC '09: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 9–16, 2009.

Julia Birke and Anoop Sarkar. A clustering approach for the nearly unsupervised recognition of nonliteral language. In *Proceedings of EACL-06*, pages 329–336, 2006.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Reseach*, 3:993–1022, 2003.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, 1998.

Ram Boukobza and Ari Rappoport. Multi-word expression identification using sentence surface features. In *Proceedings of EMNLP-09*, 2009.

Jordan Boyd-Graber and David Blei. PUTOP: turning predominant senses into a topic model for word sense disambiguation. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 277–281, 2007.

Jordan Boyd-Graber and David Blei. Syntactic topic models. In *NIPS 2008*, pages 185–192, 2008.

Jordan Boyd-Graber, David Blei, and Xiaojin Zhu. A topic model for word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1024–1033, 2007.

E. Briscoe, J. Carroll, and R. Watson. The second release of the rasp system. In *Proceedings of teh COLING/ACL 2006 Interactive Presentation Sessions*, 2006.

Samuel Brody and Mirella Lapata. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 103–111, 2009.

Alexander Budanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

Davide Buscaldi and Paolo Rosso. UPV-WSD: Combining different WSD methods by means of Fuzzy Borda Voting. In *SemEval '07: Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 434–437, 2007.

Junfu Cai, Wee Sun Lee, and Yee Whye Teh. Improving word sense disambiguation using topic features. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1015–1023, 2007.

S. Calinon. *Robot Programming by Demonstration: A Probabilistic Approach*. EPFL/CRC Press, 2009.

S. Calinon, F. Guenter, and A. Billard. On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 37(2): 286–298, 2007.

Yee Seng Chan, Hwee Tou Ng, and Zhi Zhong. NUS-PT: exploiting parallel texts for word sense disambiguation in the English all-words tasks. In *SemEval '07: Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 253–256, 2007.

Hsin-Hsi Chen, Ming-Shun Lin, and Yu-Chuan Wei. Novel association measures using web search with double checking. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1009–1016, 2006.

Rudi L. Cilibrasi and Paul M.B. Vitanyi. The Google similarity distance. *IEEE Trans. Knowledge and Data Engineering*, 19(3):370–383, 2007.

REFERENCES

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurements*, 20:37–46, 1960.

Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100–110, 1999.

Paul Cook, Afsaneh Fazly, and Suzanne Stevenson. Pulling their weight: Exploiting syntactic forms for the automatic identification of idiomatic expressions in context. In *Proceedings of the ACL-07 Workshop on A Broader Perspective on Multiword Expressions*, pages 41–48, 2007.

D. A. Cruse. Aspects of the microstructure of word meanings. In *Polysemy: Theoretical and Computational Approaches*, pages 30–51. OUP, Oxford, UK, 2000.

Mona Diab and Pravin Bhutada. Verb noun construction mwe token classification. In *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications*, pages 17–22, 2009.

Mona T. Diab and Madhav Krishna. Unsupervised classification of verb noun multi-word expression tokens. In *CICLing 2009*, pages 98–110, 2009.

Byron E. Dom. An information-theoretic external cluster-validity measure. Technical Report RJ10219, IBM, October 2001.

Afsaneh Fazly and Suzanne Stevenson. Automatically constructing a lexicon of verb phrase idiomatic combinations. In *Proceedings of EACL-06*, 2006.

Afsaneh Fazly, Paul Cook, and Suzanne Stevenson. Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics*, 35(1):61–103, 2009.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of ACL-05*, pages 363–370, 2005.

Mitchell H. Gail, Jay H. Lubin, and Lawrence V. Rubinstein. Likelihood calculations for matched case-control studies and survival studies with tied death times. *Biometrika*, 68: 703–707, 1981.

Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. In *Readings in computer vision: issues, problems, principles, and paradigms*, pages 564–584. 1987.

David Graff and Christopher Cieri. *English Gigaword*. Linguistic Data Consortium, Philadelphia, 2003.

T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(Suppl. 1):5228–5235, 2004.

Thomas L. Griffiths, Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. Integrating topics and syntax. In *Advances in Neural Information Processing Systems 17*, pages 537–544, 2005.

M.A.K. Halliday and R. Hasan. *Cohesion in English*. Longman House, New York, 1976.

J. A. Hartigan. *Clustering Algorithm*. Wiley, 1975.

Chikara Hashimoto and Daisuke Kawahara. Construction of an idiom corpus and its application to idiom identification based on WSD incorporating idiom-specific features. In *Proceedings of EMNLP-08*, pages 992–1001, 2008.

Chikara Hashimoto, Satoshi Sato, and Takehito Utsuro. Japanese idiom recognition: Drawing a line between literal and idiomatic meanings. In *Proceedings of COLING/ACL-06*, pages 353–360, 2006.

Amaç Herdağdelen, Katrin Erk, and Marco Baroni. Measuring semantic relatedness with vector space models and random walks. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 50–53, 2009.

Thomas Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57, 1999.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. Ontonotes: the 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, 2006.

Radu Ion and Dan Tufiş. Racai: meaning affinity models. In *SemEval '07: Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 282–287, 2007.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5):429–450, 2002.

Roger W. Johnson. Estimating the size of a population. *Teaching Statistics*, 16:50–52, 1994.

D. Jurafsky and J. Martin. *Speech and Language Processing*. Prentice-Hall, 2000.

Graham Katz and Eugenie Giesbrecht. Automatic identification of non-compositional multi-word expressions using latent semantic analysis. In *Proceedings of the ACL/COLING-06 Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, pages 12–19, 2006.

Ivan Kelly, James Rotton, and Roger Culver. The moon was full and nothing happened: A review of studies on the moon and human behavior. *Skeptical Inquirer*, 10(2):129–143, 1986.

Mirjam Killer, Sebastian Stüker, and Tanja Schultz. Grapheme based speech recognition. In *In Proceedings of the EUROSPEECH*, pages 3141–3144, 2003.

Beata Beigman Klebanov, Eyal Beigman, and Daniel Diermeier. Discourse topics and metaphors. In *CALC '09: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 1–8, 2009.

Ioannis Korkontzelos and Suresh Manandhar. UoY: graphs of unambiguous vertices for word sense induction and disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 355–358, 2010.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, 2001.

Anh-Cuong Le, Akira Shimazu, and Le-Minh Nguyen. Investigating problems of semi-supervised learning for word sense disambiguation. In *Proc. ICCPOL-06*, 2006.

Hang Li and Cong Li. Word translation disambiguation using bilingual bootstrapping. *Computational Linguistics*, 30:1–22, 2004.

Linlin Li. A cohesion-based approach for unsupervised recognition of literal and nonliteral use of multiword expressions. Master's thesis, Saarland University, 2008.

Linlin Li and Caroline Sporleder. Classifier combination for contextual idiom detection without labelled data. In *Proceedings of EMNLP-09*, pages 315–323, 2009.

Linlin Li and Caroline Sporleder. Linguistic cues for distinguishing literal and non-literal usage. In *Proceedings of the 23rd International Conference on Computational Linguistics (CoLing'10)*, pages 683–691, 2010a.

Linlin Li and Caroline Sporleder. Using gaussian mixture models to detect figurative language in context. In *Proceedings of the 11th Annual Conference of North American Chapter of Association for Computational Linguistics (NAACL'10), Short Papers*, pages 297–300, 2010b.

Linlin Li, Benjamin Roth, and Caroline Sporleder. Topic models for word sense disambiguation and token-based idiom detection. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, pages 1138–1147, 2010.

Dekang Lin. Automatic identification of non-compositional phrases. In *Proceedings of ACL-99*, pages 317–324, 1999.

Suresh Manandhar and Ioannis P. Klapaftis. Semeval-2010 task 14: Evaluations setting for word sense induction and disambiguation systems. In *Proceedings of the NAACL HLT Workshop on Semantic Evaluations*, 2009.

Suresh Manandhar, Ioannis P. Klapaftis, Dmitriy Dligach, and Sameer S. Pradhan. Semeval-2010 task 14: Word sense induction and disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, 2010.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Marie-Catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *LREC 2006*, 2006.

Diana McCarthy. Word sense disambiguation: An overview. *Language and Linguistics Compass*, 3(2):537–558, 2009.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 279–286, 2004.

David Mcclosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of North American ACL (NAACL)*, pages 152–159, 2006.

Marina Meila. Comparing clusterings - an information beased distance. *Journal of Multivariate Analysis*, 98:873–895, 2007.

Rada F. Mihalcea and Dan I. Moldovan. A highly accurate bootstrapping algorithm for word sense disambiguation. *Artificial Intelligence Tools*, 10(1–2):5–21, 2001.

G. Miller. Note on the bias of information estimates. *Information Theory in Psychology II-B*, pages 95–100, 1955.

George A. Miller. WordNet: a lexical database for english. *Commun. ACM*, 38(11):39–41, 1995.

Roberto Navigli. Meaningful clustering of senses helps boost word sense disambiguation performance. In *Proceedings of the 44th Annual Meeting of the Association for Computational Liguistics joint with the 21st International Conference on Computational Liguistics (COLING-ACL 2006)*, pages 105–112, 2006.

Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. SemEval-2007 Task 07: Coarse-grained English all-words task. In *Proceedings of the 4th International Workshop on Semantic Evaluation (SemEval-2007)*, pages 30–35, 2007.

Hwee Tou Ng, Daniel Chung Yong Lim, and Shou King Foo. A case study on inter-annotator agreement for word sense disambiguation. In *In Proceedings of SIGLEX Workshop On Standardizing Lexical Resources*, 1999.

Vincent Ng and Claire Cardie. Weakly supervised natural language learning without redundant views. In *Proc. of HLT-NAACL-03*, pages 94–101, 2003.

Joakim Nivre, Johan Hall, and Jens Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *Proc. of LREC-2006*, pages 2216–2219, 2006.

Liam Paninski. Estimation of entropy and mutual information. *Neural Computation*, 15: 1191–1253, 2003.

Sung Y. Park and Anil K. Bera. Maximum entropy autoregressive conditional heteroskedasticity model. *Journal of Econometrics*, 150(2):219–230, 2009.

Ted Pedersen. Umnd2: Senseclusters applied to the sense induction task of senseval-4. In *Proceedings of the 4th International Workshop on Semantic Evaluation*, pages 394–397, 2007.

Ted Pedersen. Duluth-wsi: Senseclusters applied to the sense induction task of semeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 363–366, 2010.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. Wordnet::similarity: measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, pages 38–41, 2004.

M.F. Porter. Snowball: A language for stemming algorithms. `http://snowball.tartarus.org/texts/introduction.html`, October 2001.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. SemEval-2007 Task-17: English lexical sample, srl and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluation (SemEval-2007)*, pages 87–92, 2007.

Amruta Purandare and Ted Pedersen. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the CoNLL*, pages 41–48, 2004.

M. Quenouille. Notes on bias and estimation. *Biometrika*, 43:353–360, 1956.

Adwait Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proceedings of EMNLP-96*, 1996.

Susanne Riehemann. *A Constructional Approach to Idioms and Word Formation*. PhD thesis, Stanford University, 2001.

Ellen Riloff and Jessica Shepherd. A corpus-based approach for building semantic lexicons. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 117–124, 1997.

Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 EMNLP-CoNll Joint Conference*, pages 410–420, 2007.

Benjamin Roth and Dietrich Klakow. Combining wikipedia-based concept models for cross-language retrieval. In *Proceedings of the Information Retrieval Facility Conference*, 2010.

Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Commun. ACM*, 8:627–633, 1965.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. Multiword expressions: a pain in the neck for NLP. In *Lecture Notes in Computer Science*, 2001.

Hinrich Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1): 97–123, 1998.

C. E. Shannon and W. Weaver. *The mathematical theory of communication*. Univ. of Ill. Press, 1998.

Fei Song and W. Bruce Croft. A general language model for information retrieval (poster abstract). In *Research and Development in Information Retrieval*, pages 279–280, 1999.

Philipp Sorg and Philipp Cimiano. Cross-lingual information retrieval with explicit semantic analysis. In *Working Notes for the CLEF 2008 Workshop*, 2008.

Caroline Sporleder and Linlin Li. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In *Proceedings of EACL-09*, 2009.

Caroline Sporleder, Linlin Li, and Alexis Palmer. Cohesive links with literal idiomatic expressions in discourse: an empirical and computational study. *Multidisciplinary Approaches to Discourse 2010*, to appear.

Alexander Strehl and Joydeep Gosh. Cluster ensembles: A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

S. Strong, R. Koberle, S. R. van de Ruyter, and W. Bialek. Entropy and information in neural spike trains. *Physical Review Letters*, 80:197–202, 1998.

J. Tukey. Bias and confidence in not quite large samples. *Annals of Mathematical Statistics*, 29: 614, 1958.

Jean Véronis. Hyperlex: Lexical cartography for information retrieval. *Computer Speech and Language*, 18(3):223–252, 2004.

David Vickrey, Luke Biewald, Marc Teyssier, and Daphne Koller. Word-sense disambiguation for machine translation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 771–778, 2005.

Aline Villavicencio, Valia Kordoni, Yi Zhang, Marco Idiart, and Carlos Ramisch. Validation and evaluation of automatically acquired multiword expressions for grammar engineering. In *Proceedings of EMNLP-07*, pages 1034–1043, 2007.

Y. H. Wang. On the number of successes in independent trials. *Statistica Sinica 3*, 2:295–312, 1993.

Yi Wang, Hongjie Bai, Matt Stanton, Wen-Yen Chen, and Edward Y. Chang. Plda: Parallel latent dirichlet allocation for large-scale applications. In *Proc. of 5th International Conference on Algorithmic Aspects in Information and Management*, 2009.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196, 1995.

Y Zhao and G Karypis. Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery*, 10(2):141–168, 2005.

G. K. Zipf. *The psychobiology of language*. Houghton Mifflin, Oxford, England, 1935.